IBM zSystems and LinuxONE

Hardware Management Console Web Services API Version 2.16.0

SC27-2642-02

Note:

Before you use this information and the product it supports, read the information in "Safety" on page lxix, Appendix E, "Notices," on page 1453, and *IBM Systems Environmental Notices and User Guide*, Z125-5823.

This edition, SC27-2642-02, applies to the IBM Z and IBM LinuxONE servers. This edition replaces SC27-2642-01.

There might be a newer version of this document in a **PDF** file available on **IBM Documentation**. Go to https://www.ibm.com/docs/en/systems-hardware, select **IBM Z** or **IBM LinuxONE**, then select your configuration, and click **Library Overview** on the navigation bar.

© Copyright International Business Machines Corporation 2022, 2024.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Figures ..... **xv**

Tables ..... **xlv**

Safety ..... **lxix**
Safety notices lxix
World trade safety information lxix
Laser safety information lxix
Laser compliance lxix

About this publication ..... **lxxi**
Related publications lxxi
Related HMC and SE console information lxxi
Revisions lxxi
Accessibility features lxxi
Consult assistive technologies lxxii
Keyboard navigation lxxii
IBM and accessibility lxxii
How to provide feedback to IBM lxxii

Part 1. Web Services API fundamentals **1**

Chapter 1. Introduction 3
Overview 3
Components of the API 3
Enabling and accessing the API 5
Authentication and access control 5
Compatibility 6
API versioning 6
Allowable changes within a major version 6
Requirements on client applications 7
Summary of API version updates 7

Chapter 2. Base definitions 55
Data types 55
Input and output representation 57
Representing API data types in JSON 57

Chapter 3. Invoking API operations 59
HTTP protocol standard 59
Connecting to the API HTTP server 59
HTTP header field usage 59
Required request header fields 60
Optional request headers 61
Standard response headers 62
Additional response headers 62
Media types 63
HTTP status codes 63
Error response bodies 65

Common request validation reason codes....66
Common request processing reason codes....72
Use of chunked response encoding....74
Filter query parameters....74
Regular expression syntax....75

Chapter 4. Asynchronous notification....77
Grouping of notifications....77
Java Message Service (JMS) basics....78
Connecting to the API message broker....78
Per-session notification topics....79
Notification message formats....80
Server-Sent Events (SSE)....90
Security considerations....90
Initial events....90
Notification event formats....91

Chapter 5. Data model definitions....97
Data model concepts....97
Objects in the data model....97
Properties in the data model....98
Shared data model schema elements....100
Base managed object properties schema....100

Chapter 6. Features....103
API features....103
Firmware features....104
dpm-storage-management....105
dpm-fcp-tape-management....106
dpm-smcd-partition-link-management....107

Part 2. General services....109

Chapter 7. General API services....111
General API services operations summary....111
Session management services....112
Query API Version....113
Logon....115
Establish Shared Secret Key....121
Provide Requested MFA Information....123
Change Logon Password....126
Verify Logon Password....129
Logoff....130
Get Notification Topics....131
Create Server-Sent Events Stream....134
Update Server-Sent Events Stream....139
Delete Server-Sent Events Stream....140
Open Server-Sent Events Stream....141
Get Server-Sent Events Stream Last Event ID....143
Request aggregation services....144
Submit Requests....144
Asynchronous job processing....151
Query Job Status....151
Delete Completed Job Status....154
Cancel Job....156

Chapter 8. Inventory and metrics services....159

Inventory services operations summary....159
Metrics service operations summary....159
Inventory service....160
Get Inventory....160
Metrics service....169
Create Metrics Context....169
Get Metrics....172
Delete Metrics Context....175

Chapter 9. Metric groups....177
Monitors dashboard metric groups....177
Channels....177
CPC overview....178
DPM system overview....179
Logical partitions....180
Partitions....181
zCPC environmentals and power....182
Power status....183
zCPC processors....185
Cryptos....186
Adapters....186
Flash memory adapters....187
RoCE adapters....187
Network management metrics....188
Network adapter port metric group....188
Network interface metric group....190

Part 3. CPC management....193

Chapter 10. Dynamic Partition Manager (DPM)....195
FICON storage configuration....195
Device number constraints....197
Operations summary....197
Partition operations summary....197
Adapter operations summary....200
Virtual Switch operations summary....201
Capacity Group operations summary....201
Storage Site operations summary....202
Storage Fabric operations summary....202
Storage Switch operations summary....203
Storage Subsystem operations summary....203
Storage Control Unit operations summary....204
Storage Group operations summary....205
Storage Template operations summary....206
Tape Library operations summary....207
Tape Link operations summary....207
Partition Link operations summary....208
Partition object....209
Data model....209
List Partitions of a CPC....234
List Permitted Partitions....237
Create Partition....239
Delete Partition....245
Delete Partition Asynchronously....247
Get Partition Properties....250
Update Partition Properties....253
Update Partition Properties Asynchronously....257

Start Partition....262
Attach Storage Group to Partition....266
Stop Partition....269
Dump Partition....271
Start Dump Program....275
Perform PSW Restart....281
Create Virtual Function....283
Delete Virtual Function....286
Get Virtual Function Properties....287
Update Virtual Function Properties....289
Create NIC....291
Delete NIC....296
Get NIC Properties....298
Update NIC Properties....300
Increase Crypto Configuration....305
Change Crypto Domain Configuration....308
Decrease Crypto Configuration....310
Zeroize Crypto Domain....313
Mount ISO Image....316
Unmount ISO Image....317
Detach Storage Group from Partition....319
Create HBA....321
Delete HBA....324
Update HBA Properties....326
Get HBA Properties....328
Reassign Storage Adapter Port....330
Send OS Command....332
Open OS Message Channel....334
List OS Messages of a Partition....336
Delete Partition OS Message....339
Get ASCII Console WebSocket URI....341
Attach Tape Link to Partition....343
Detach Tape Link from Partition....346
Report a Partition Problem....348
Get Partition Historical Sustainability Data....350
Assign Certificate to Partition....353
Unassign Certificate from Partition....355
Inventory service data....357
Adapter object....360
Data model....360
List Adapters of a CPC....374
List Permitted Adapters....377
Get Adapter Properties....381
Update Adapter Properties....383
Change Crypto Type....386
Create Hipersocket....388
Delete Hipersocket....390
Get Partitions Assigned to Adapter....392
Get Network Port Properties....394
Update Network Port Properties....396
Get Storage Port Properties....398
Update Storage Port Properties....399
Change Adapter Type....401
Update Adapter Firmware....403
Inventory service data....407
Virtual Switch object....408
Data model....408
List Virtual Switches of a CPC....409

List Permitted Virtual Switches.... 411
Get Virtual Switch Properties.... 414
Get Connected VNICs of a Virtual Switch.... 416
Update Virtual Switch Properties.... 417
Inventory service data.... 419
Capacity Group element object.... 420
Data model.... 420
List Capacity Groups of a CPC.... 421
Create Capacity Group.... 423
Delete Capacity Group.... 426
Get Capacity Group Properties.... 427
Add Partition to Capacity Group.... 429
Remove Partition from Capacity Group.... 431
Update Capacity Group Properties.... 433
Inventory service data.... 435
Storage Site object.... 436
Data model.... 436
List Storage Sites.... 437
Create Storage Site.... 439
Delete Storage Site.... 442
Get Storage Site Properties.... 444
Update Storage Site Properties.... 445
Inventory service data.... 447
Storage Fabric object.... 448
Data model.... 448
List Storage Fabrics.... 449
Create Storage Fabric.... 451
Delete Storage Fabric.... 454
Get Storage Fabric Properties.... 455
Update Storage Fabric Properties.... 457
Inventory service data.... 459
Storage Switch object.... 460
Data model.... 460
List Storage Switches of a Storage Site.... 461
List Storage Switches of a Storage Fabric.... 463
Define Storage Switch.... 466
Undefine Storage Switch.... 468
Get Storage Switch Properties.... 470
Update Storage Switch Properties.... 471
Move Storage Switch to Storage Site.... 473
Move Storage Switch to Storage Fabric.... 475
Inventory service data.... 477
Storage Subsystem object.... 478
Data model.... 478
List Storage Subsystems of a Storage Site.... 480
Define Storage Subsystem.... 482
Undefine Storage Subsystem.... 484
Get Storage Subsystem Properties.... 486
Update Storage Subsystem Properties.... 488
Move Storage Subsystem to Storage Site.... 489
Add Connection Endpoint.... 491
Remove Connection Endpoint.... 494
Inventory service data.... 496
Storage Control Unit object.... 497
Data model.... 497
List Storage Control Units of a Storage Subsystem.... 500
Define Storage Control Unit.... 502
Undefine Storage Control Unit.... 504

Get Storage Control Unit Properties....506
Update Storage Control Unit Properties.... 507
Add Volume Range.... 509
Remove Volume Range....511
Create Storage Path....513
Delete Storage Path....516
Get Storage Path Properties....518
Update Storage Path Properties....519
Inventory service data....522
Storage Group object.... 523
Data model....525
List Storage Groups.... 544
Create Storage Group.... 547
Delete Storage Group.... 553
Get Storage Group Properties.... 556
Modify Storage Group Properties....558
Resend Request....568
Add Candidate Adapter Ports to an FCP Storage Group....571
Remove Candidate Adapter Ports from an FCP Storage Group.... 573
List Storage Volumes of a Storage Group....575
Get Storage Volume Properties....578
Fulfill FICON Storage Volume....580
Fulfill FICON Storage Volumes....583
Fulfill FCP Storage Volume.... 587
Accept Mismatched Storage Volumes.... 589
Reject Mismatched FCP Storage Volumes.... 591
List Virtual Storage Resources of a Storage Group....594
Get Virtual Storage Resource Properties.... 597
Update Virtual Storage Resource Properties....598
Get Partitions for a Storage Group.... 601
Validate LUN Path.... 603
Start FCP Storage Discovery....605
Get Connection Report.... 608
Get Storage Group Histories....617
Inventory service data....632
Storage Template object....635
Data model....635
List Storage Templates....642
Create Storage Template....644
Delete Storage Template....647
Get Storage Template Properties....649
Modify Storage Template Properties....650
List Storage Template Volumes of a Storage Template....656
Get Storage Template Volume Properties.... 659
Inventory service data....660
Tape Library object....661
Data model....662
List Tape Libraries....663
Undefine Tape Library....665
Get Tape Library Properties....666
Update Tape Library Properties.... 668
Request Tape Library Zoning....670
Discover Tape Libraries.... 672
Inventory service data....675
Tape Link object.... 676
Data model....677
List Tape Links....683
Create Tape Link.... 685

Get Tape Link Properties.... 689
Modify Tape Link Properties.... 691
Delete Tape Link.... 695
Add Adapter Ports.... 697
Remove Adapter Ports.... 700
Replace Adapter Port.... 702
Resend Request.... 704
List Virtual Tape Resources of a Tape Link.... 707
Get Virtual Tape Resource Properties.... 709
Update Virtual Tape Resource Properties.... 710
Get Partitions for a Tape Link.... 713
Get Tape Link Histories.... 715
Update Tape Link Environment Report.... 726
Get Tape Link Environment Report.... 729
Inventory service data.... 734
Partition Link object.... 736
Data model.... 736
Create Partition Link.... 742
Delete Partition Link.... 755
Get Partition Link Properties.... 759
List Partition Links.... 762
Modify Partition Link.... 764
Inventory service data.... 779

Chapter 11. Core IBM zSystems resources.... 781
Operations Summary.... 781
Console operations summary.... 781
User operations summary.... 783
User Role operations summary.... 784
Task operations summary.... 784
User Pattern operations summary.... 785
Password Rule operations summary.... 785
LDAP Server Definition operations summary.... 786
MFA Server Definition operations summary.... 786
Group operations summary.... 787
CPC operations summary.... 788
Logical partition operations summary.... 791
Certificate operations summary.... 792
Activation profile operations summary.... 793
Capacity record operations summary.... 795
Adapter operations summary.... 795
Shared nested objects.... 796
Console object.... 801
Data model.... 801
Get Console Properties.... 812
Restart Console.... 819
Shutdown Console.... 821
Reorder User Patterns.... 822
Get Console Audit Log.... 824
Get Console Security Log.... 830
Get Console Events Log.... 833
List Console Hardware Messages.... 839
Get Console Hardware Message Properties.... 841
Delete Console Hardware Message.... 843
Request Console Service.... 844
Get Console Service Request Information.... 846
Decline Console Service.... 848
List Unmanaged CPCs.... 850

Get Mobile App Preferences....852
Set Mobile App Preferences....853
Get CPC Notification Preferences for Device....856
Update CPC Notification Preferences for Device....860
List Remote Firmware Updates of the Console....864
Get Console Remote Firmware Update Properties....867
Delete Console Remote Firmware Update....868
Authorize Remote Firmware Updates....870
Update Welcome Text....871
Get Console Notification Preferences for Device....873
Update Console Notification Preferences for Device....875
Console Single Step Install....876
Report a Console Problem....882
Console Delete Retrieved Internal Code....884
List Console API Features....887
Inventory service data....888
User-related-access permission....892
User object....893
Data model....893
List Users....900
Get User Properties....902
Update User Properties....904
Add User Role to User....907
Remove User Role from User....909
Create User....911
Delete User....915
Inventory service data....917
User Role object....918
Data model....918
List User Roles....921
Get User Role Properties....923
Update User Role Properties....925
Add Permission to User Role....927
Remove Permission from User Role....930
Create User Role....933
Delete User Role....935
Inventory service data....936
Task object....937
Data model....937
List Tasks....938
Get Task Properties....940
Inventory service data....941
User Pattern object....942
Data model....942
List User Patterns....946
Get User Pattern Properties....947
Update User Pattern Properties....949
Create User Pattern....952
Delete User Pattern....955
Inventory service data....957
Password Rule object....957
Data model....958
List Password Rules....961
Get Password Rule Properties....963
Update Password Rule Properties....965
Create Password Rule....967
Delete Password Rule....969
Inventory service data....970

LDAP Server Definition object....971
Data model....972
List LDAP Server Definitions....977
Get LDAP Server Definition Properties....978
Update LDAP Server Definition Properties....980
Create LDAP Server Definition....982
Delete LDAP Server Definition....984
Inventory service data....986
MFA Server Definition object....987
Data model....987
List MFA Server Definitions....988
Get MFA Server Definition Properties....990
Update MFA Server Definition Properties....991
Create MFA Server Definition....993
Delete MFA Server Definition....995
Inventory service data....996
Group Object....996
Data model....997
List Custom Groups....998
Get Custom Group Properties....1000
Create Custom Group....1002
Delete Custom Group....1004
Add Member to Custom Group....1005
Remove Member from Custom Group....1007
List Custom Group Members....1008
Inventory service data....1010
CPC object....1010
Data model....1010
List CPC Objects....1036
Get CPC Properties....1039
Update CPC Properties....1048
Start CPC....1049
Stop CPC....1052
Activate CPC....1054
Deactivate CPC....1057
Import Profiles....1059
Export Profiles....1060
Set Auto-Start List....1062
Add Temporary Capacity....1064
Remove Temporary Capacity....1067
Swap Current Time Server....1069
Set STP Configuration....1071
Change STP-only Coordinated Timing Network....1074
Join STP-only Coordinated Timing Network....1075
Leave STP-only Coordinated Timing Network....1077
Get CPC Audit Log....1078
Get CPC Security Log....1081
Get CPC Events Log....1084
List CPC Hardware Messages....1088
Get CPC Hardware Message Properties....1091
Delete CPC Hardware Message....1093
Request CPC Service....1095
Get CPC Service Request Information....1097
Decline CPC Service....1100
Export WWPN List....1101
Import DPM Configuration....1104
List Remote Firmware Updates of a CPC....1115
Get CPC Remote Firmware Update Properties....1117

Delete CPC Remote Firmware Update....1119
Get Logical Partition Resource Assignments....1121
Get LPAR Controls....1122
Update LPAR Controls....1134
CPC Single Step Install....1136
Import Secure Execution Key....1142
Delete Secure Execution Key....1145
Import CPC Certificate....1147
Report a CPC Problem....1149
Get CPC Historical Sustainability Data....1152
CPC Install and Activate....1157
CPC Delete Retrieved Internal Code....1162
List CPC API Features....1165
Switch Support Elements....1167
Inventory service data....1169
Logical Partition object....1169
Data model....1169
List Logical Partitions of CPC....1194
List Permitted Logical Partitions....1196
Get Logical Partition Properties....1200
Update Logical Partition Properties....1205
Activate Logical Partition....1207
Deactivate Logical Partition....1212
Reset Normal....1214
Reset Clear....1216
Load....1218
Load Logical Partition....1224
Load Logical Partition from FTP....1227
PSW Restart....1230
Start Logical Partition....1232
Stop Logical Partition....1233
Send OS Command....1235
Open OS Message Channel....1237
List OS Messages of a Logical Partition....1239
Delete Logical Partition OS Message....1243
SCSI Load....1245
SCSI Dump....1248
NVMe Load....1251
NVMe Dump....1254
Assign Certificate to Logical Partition....1256
Unassign Certificate from Logical Partition....1258
Report a Logical Partition Problem....1260
Get Logical Partition Historical Sustainability Data....1263
Inventory service data....1266
Certificate object....1266
Data model....1266
Delete Certificate....1267
Get Certificate Properties....1268
Get Encoded Certificate....1271
List Certificates....1273
Update Certificate Properties....1275
Inventory service data....1277
Reset activation profile....1278
Data model....1278
List Reset Activation Profiles....1279
Get Reset Activation Profile Properties....1281
Update Reset Activation Profile Properties....1283
Create Reset Activation Profile....1285

Delete Reset Activation Profile.... 1289
Inventory service data....1291
Image activation profile....1291
Data model.... 1291
List Image Activation Profiles.... 1307
Get Image Activation Profile Properties.... 1310
Update Image Activation Profile Properties.... 1315
Assign Certificate to Image Activation Profile.... 1316
Unassign Certificate from Image Activation Profile.... 1319
Create Image Activation Profile.... 1321
Delete Image Activation Profile.... 1338
Inventory service data....1340
Load activation profile.... 1340
Data model.... 1340
List Load Activation Profiles.... 1346
Get Load Activation Profile Properties....1349
Update Load Activation Profile Properties....1351
Create Load Activation Profile....1353
Delete Load Activation Profile....1360
Inventory service data....1362
Group profile.... 1362
Data model.... 1362
List Group Profiles.... 1364
Get Group Profile Properties.... 1366
Update Group Profile Properties....1369
Create Group Profile.... 1370
Delete Group Profile.... 1373
Inventory service data....1375
Capacity records.... 1375
Data model.... 1375
List Capacity Records.... 1378
Get Capacity Record Properties....1380
Inventory service data....1381

Chapter 12. Energy management.... 1383
Groups....1384
Special states....1385
Power saving.... 1386
Group power saving....1386
Power capping....1386
Group capping.... 1386
Energy management operations summary.... 1387
Energy Management for CPC object....1387
Data model.... 1388
Operations.... 1388
Set CPC Power Save.... 1388
Set CPC Power Capping....1390
Set zCPC Power Save....1393
Set zCPC Power Capping.... 1395
Get CPC Energy Management Data....1397
Get Energy Optimization Advice Summary....1399
Get Energy Optimization Advice Details.... 1402

Appendix A. Base Control Program internal interface (BCPii).... 1411
Security controls....1425
Special considerations for BCPii.... 1425
Configuring the Support Element for BCPii....1427

Asynchronous notification support....1428
Notification Registration....1429
Security....1434

Appendix B. Enum values for a type of managed objects within User Roles....1437

Appendix C. Enum values for the User Role object....1439

Appendix D. Enum values for the Task object....1441

Appendix E. Notices....1453
Trademarks....1453
Class A Notices....1454

Index....1459

Figures

1. Query API Version: Request 114

2. Query API Version: Response 115

3. Logon: Request 121

4. Logon: Response 121

5. Establish Shared Secret Key: Request 123

6. Establish Shared Secret Key: Response 123

7. Provide Requested MFA Information: Request 126

8. Provide Requested MFA Information: Response 126

9. Change Logon Password: Request 128

10. Change Logon Password: Response 128

11. Verify Logon Password: Request 130

12. Verify Logon Password: Response 130

13. Logoff: Request 131

14. Logoff: Response 131

15. Get Notification Topics: Request 133

16. Get Notification Topics: Response 133

17. Create Server-Sent Events Stream: Request 138

18. Create Server-Sent Events Stream: Response 138

19. Update Server-Sent Events Stream: Request 140

20. Update Server-Sent Events Stream: Response 140

21. Delete Server-Sent Events Stream: Request 141

22. Delete Server-Sent Events Stream: Response 141

23. Get Server-Sent Events Stream Last Event ID: Request 144

24. Get Server-Sent Events Stream Last Event ID: Response....144

25. Submit Requests: Request....150

26. Submit Requests: Response....150

27. Query Job Status: Request....154

28. Query Job Status: Response....154

29. Delete Completed Job Status: Request....155

30. Delete Completed Job Status: Response....156

31. Cancel Job: Request....157

32. Cancel Job: Response....157

33. Get Inventory: Request....164

34. Get Inventory: Response (Part 1)....165

35. Get Inventory: Response (Part 2)....166

36. Get Inventory: Response (Part 3)....167

37. Get Inventory: Response (Part 4)....168

38. Create Metrics Context: Request....171

39. Create Metrics Context: Response....172

40. Get Metrics: Request....175

41. Get Metrics: Response....175

42. Delete Metrics Context: Request....176

43. Delete Metrics Context: Response....176

44. List Partitions of a CPC: Request....236

45. List Partitions of a CPC: Response....236

46. List Permitted Partitions: Request....239

47. List Permitted Partitions: Response....239

48. Create Partition: Request....245

49. Create Partition: Response....245

50. Delete Partition: Request....247

51. Delete Partition: Response....247

52. Delete Partition Asynchronously: Request....249

53. Delete Partition Asynchronously: Response....250

54. Get Partition Properties: Request....251

55. Get Partition Properties: Response (Part 1)....252

56. Get Partition Properties: Response (Part 2)....253

57. Update Partition Properties: Request....257

58. Update Partition Properties: Response....257

59. Update Partition Properties Asynchronously: Request....262

60. Update Partition Properties Asynchronously: Response....262

61. Start Partition: Request....266

62. Start Partition: Response....266

63. Attach Storage Group to Partition: Request....268

64. Attach Storage Group to Partition: Response....269

65. Stop Partition: Request....271

66. Stop Partition: Response....271

67. Dump Partition: Request....274

68. Dump Partition: Response....275

69. Start Dump Program: Request....281

70. Start Dump Program: Response....281

71. Perform PSW Restart: Request....283

72. Perform PSW Restart: Response....283

73. Create Virtual Function: Request....285

74. Create Virtual Function: Response.....286

75. Delete Virtual Function: Request.....287

76. Delete Virtual Function: Response.....287

77. Get Virtual Function Properties: Request.....289

78. Get Virtual Function Properties: Response.....289

79. Update Virtual Function Properties: Request.....291

80. Update Virtual Function Properties: Response.....291

81. Create NIC: Request.....296

82. Create NIC: Response.....296

83. Delete NIC: Request.....298

84. Delete NIC: Response.....298

85. Get NIC Properties: Request.....300

86. Get NIC Properties: Response.....300

87. Update NIC Properties: Request.....305

88. Update NIC Properties: Response.....305

89. Increase Crypto Configuration: Request.....307

90. Increase Crypto Configurations: Response.....308

91. Change Crypto Domain Configuration: Request.....310

92. Change Crypto Domain Configuration: Response.....310

93. Decrease Crypto Configuration: Request.....313

94. Decrease Crypto Configuration: Response.....313

95. Zeroize Crypto Domain: Request.....315

96. Zeroize Crypto Domain: Response.....315

97. Mount ISO Image: Request.....317

98. Mount ISO Image: Response.....317

99. Unmount ISO Image: Request..... 319

100. Unmount ISO Image: Response.....319

101. Detach Storage Group from Partition: Request..... 321

102. Detach Storage Group from Partition: Response.....321

103. Create HBA: Request.....323

104. Create HBA: Response..... 324

105. Delete HBA: Request..... 325

106. Delete HBA: Response.....326

107. Update HBA Properties: Request.....328

108. Update HBA Properties: Response..... 328

109. Get HBA Properties: Request.....329

110. Get HBA Properties: Response..... 330

111. Reassign Storage Adapter Port: Request.....332

112. Reassign Storage Adapter Port: Response..... 332

113. Send OS Command: Request..... 333

114. Send OS Command: Response.....333

115. Open OS Message Channel: Request.....335

116. Open OS Message Channel: Response..... 335

117. List OS Messages of a Partition: Request.....338

118. List OS Messages of a Partition: Response..... 339

119. Delete Partition OS Message: Request..... 340

120. Delete Partition OS Message: Response.....341

121. Get ASCII Console WebSocket URI: Request.....343

122. Get ASCII Console WebSocket URI: Response..... 343

123. Attach Tape Link to Partition: Request.....345

124. Attach Tape Link to Partition: Response..... 345

125. Detach Tape Link from Partition: Request..... 347

126. Detach Tape Link from Partition: Response..... 348

127. Report a Partition Problem: Request..... 350

128. Report a Partition Problem: Response..... 350

129. Get Partition Historical Sustainability Data: Request..... 353

130. Get Partition Historical Sustainability Data: Response..... 353

131. Assign Certificate to Partition: Request..... 355

132. Assign Certificate to Partition: Response..... 355

133. Unassign Certificate from Partition: Request..... 356

134. Unassign Certificate from Partition: Response..... 357

135. Partition object: Sample inventory data - Response (Part 1)..... 358

136. Partition object: Sample inventory data - Response (Part 2)..... 359

137. Partition object: Sample inventory data - Response (Part 3)..... 359

138. List Adapters of a CPC: Request..... 376

139. List Adapters of a CPC: Response..... 377

140. List Permitted Adapters: Request..... 380

141. List Permitted Adapters: Response..... 381

142. Get Adapter Properties: Request..... 382

143. Get Adapter Properties: Response..... 383

144. Update Adapter Properties: Request..... 385

145. Update Adapter Properties: Response..... 385

146. Change Crypto Type: Request..... 387

147. Change Crypto Type: Response..... 388

148. Create Hipersocket: Request..... 390

149. Create Hipersocket: Response....390

150. Delete Hipersocket: Request....392

151. Delete Hipersocket: Response....392

152. Get Partitions Assigned to Adapter: Request....394

153. Get Partitions Assigned to Adapter: Response....394

154. Get Network Port Properties: Request....395

155. Get Network Port Properties: Response....396

156. Update Network Port Properties: Request....397

157. Update Network Port Properties: Response....398

158. Get Storage Port Properties: Request....399

159. Get Storage Port Properties: Response....399

160. Update Storage Port Properties: Request....401

161. Update Storage Port Properties: Response....401

162. Change Adapter Type: Request....403

163. Change Adapter Type: Response....403

164. Update Adapter Firmware: Request....406

165. Update Adapter Firmware: Response....406

166. Adapter object: Sample inventory data....407

167. List Virtual Switches of a CPC: Request....411

168. List Virtual Switches of a CPC: Response....411

169. List Permitted Virtual Switches: Request....413

170. List Permitted Virtual Switches: Response....414

171. Get Virtual Switch Properties: Request....415

172. Get Virtual Switch Properties: Response....416

173. Get Connected VNICs of a Virtual Switch: Request....417

174. Get Connected VNICs of a Virtual Switch: Response..... 417

175. Update Virtual Switch Properties: Request..... 419

176. Update Virtual Switch Properties: Response..... 419

177. Virtual Switch object: Sample inventory data - Response..... 419

178. List Capacity Groups of a CPC: Request..... 423

179. List Capacity Groups of a CPC: Response..... 423

180. Create Capacity Group: Request..... 425

181. Create Capacity Group: Response..... 426

182. Delete Capacity Group: Request..... 427

183. Delete Capacity Group: Response..... 427

184. Get Capacity Group Properties: Request..... 429

185. Get Capacity Group Properties: Response..... 429

186. Add Partition to Capacity Group: Request..... 431

187. Add Partition to Capacity Group: Response..... 431

188. Remove Partition from Capacity Group: Request..... 433

189. Remove Partition from Capacity Group: Response..... 433

190. Update Capacity Group Properties: Request..... 435

191. Update Capacity Group Properties: Response..... 435

192. List Storage Sites: Request..... 439

193. List Storage Sites: Response..... 439

194. Create Storage Site: Request..... 441

195. Create Storage Site: Response..... 442

196. Delete Storage Site: Request..... 443

197. Delete Storage Site: Response..... 443

198. Get Storage Site Properties: Request..... 445

199. Get Storage Site Properties: Response....445

200. Update Storage Site Properties: Request.... 447

201. Update Storage Site Properties: Response....447

202. Storage Site object: Sample inventory data - Response.... 448

203. List Storage Fabrics: Request....451

204. List Storage Fabrics: Response.... 451

205. Create Storage Fabric: Request....453

206. Create Storage Fabric: Response.... 454

207. Delete Storage Fabric: Request....455

208. Delete Storage Fabric: Response.... 455

209. Get Storage Fabric Properties: Request....457

210. Get Storage Fabric Properties: Response.... 457

211. Update Storage Fabric Properties: Request....459

212. Update Storage Fabric Properties: Response....459

213. Storage Fabric object: Sample inventory data - Response....459

214. List Storage Switches of a Storage Site: Request.... 463

215. List Storage Switches of a Storage Site: Response....463

216. List Storage Switches of a Storage Fabric: Request.... 465

217. List Storage Switches of a Storage Fabric: Response....465

218. Define Storage Switch: Request....468

219. Define Storage Switch: Response.... 468

220. Undefine Storage Switch: Request....469

221. Undefine Storage Switch: Response....470

222. Get Storage Switch Properties: Request....471

223. Get Storage Switch Properties: Response....471

224. Update Storage Switch Properties: Request....473

225. Update Storage Switch Properties: Response....473

226. Move Storage Switch to Storage Site: Request....475

227. Move Storage Switch to Storage Site: Response....475

228. Move Storage Switch to Storage Fabric: Request....477

229. Move Storage Switch to Storage Fabric: Response....477

230. Storage Switch object: Sample inventory data - Response....478

231. List Storage Subsystems of a Storage Site: Request....481

232. List Storage Subsystems of a Storage Site: Response....482

233. Define Storage Subsystem: Request....484

234. Define Storage Subsystem: Response....484

235. Undefine Storage Subsystem: Request....485

236. Undefine Storage Subsystem: Response....486

237. Get Storage Subsystem Properties: Request....487

238. Get Storage Subsystem Properties: Response....487

239. Update Storage Subsystem Properties: Request....489

240. Update Storage Subsystem Properties: Response....489

241. Move Storage Subsystem to Storage Site: Request....491

242. Move Storage Subsystem to Storage Site: Response....491

243. Add Connection Endpoint: Request....494

244. Add Connection Endpoint: Response....494

245. Remove Connection Endpoint: Request....496

246. Remove Connection Endpoint: Response....496

247. Storage Subsystem object: Sample inventory data - Response....497

248. List Storage Control Units of a Storage Subsystem: Request....501

249. List Storage Control Units of a Storage Subsystem: Response..... 502

250. Define Storage Control Unit: Request..... 504

251. Define Storage Control Unit: Response..... 504

252. Undefine Storage Control Unit: Request..... 505

253. Undefine Storage Control Unit: Response..... 505

254. Get Storage Control Unit Properties: Request..... 507

255. Get Storage Control Unit Properties: Response..... 507

256. Update Storage Control Unit Properties: Request..... 509

257. Update Storage Control Unit Properties: Response..... 509

258. Add Volume Range: Request..... 511

259. Add Volume Range: Response..... 511

260. Remove Volume Range: Request..... 513

261. Remove Volume Range: Response..... 513

262. Create Storage Path: Request..... 516

263. Create Storage Path: Response..... 516

264. Delete Storage Path: Request..... 518

265. Delete Storage Path: Response..... 518

266. Get Storage Path Properties: Request..... 519

267. Get Storage Path Properties: Response..... 519

268. Update Storage Path Properties: Request..... 522

269. Update Storage Path Properties: Response..... 522

270. Storage Control Unit object: Sample inventory data - Response..... 523

271. List Storage Groups: Request..... 546

272. List Storage Groups: Response..... 547

273. Create Storage Group: Request..... 553

274. Create Storage Group: Response..... 553

275. Delete Storage Group: Request..... 556

276. Delete Storage Group: Response..... 556

277. Get Storage Group Properties: Request..... 557

278. Get Storage Group Properties: Response..... 558

279. Modify Storage Group Properties: Request..... 568

280. Modify Storage Group Properties: Response..... 568

281. Resend Request: Request..... 570

282. Resend Request: Response..... 571

283. Add Candidate Adapter Ports to an FCP Storage Group: Request..... 573

284. Add Candidate Adapter Ports to an FCP Storage Group: Response..... 573

285. Remove Candidate Adapter Ports from an FCP Storage Group: Request..... 575

286. Remove Candidate Adapter Ports from an FCP Storage Group: Response..... 575

287. List Storage Volumes of a Storage Group: Request..... 577

288. List Storage Volumes of a Storage Group: Response..... 578

289. Get Storage Volume Properties: Request..... 579

290. Get Storage Volume Properties: Response..... 580

291. Fulfill FICON Storage Volume: Request..... 583

292. Fulfill FICON Storage Volume: Response..... 583

293. Fulfill FICON Storage Volumes: Request..... 586

294. Fulfill FICON Storage Volumes: Response..... 586

295. Fulfill FCP Storage Volume: Request..... 589

296. Fulfill FCP Storage Volume: Response..... 589

297. Accept Mismatched Storage Volumes: Request..... 591

298. Accept Mismatched Storage Volumes: Response..... 591

299. Reject Mismatched FCP Storage Volumes: Request....594

300. Reject Mismatched FCP Storage Volumes: Response....594

301. List Virtual Storage Resources of a Storage Group: Request....596

302. List Virtual Storage Resources of a Storage Group: Response....596

303. Get Virtual Storage Resource Properties: Request....598

304. Get Virtual Storage Resource Properties: Response....598

305. Update Virtual Storage Resource Properties: Request....600

306. Update Virtual Storage Resource Properties: Response....601

307. Get Partitions for a Storage Group: Request....602

308. Get Partitions for a Storage Group: Response....603

309. Validate LUN Path: Request....605

310. Validate LUN Path: Response....605

311. Start FCP Storage Discovery: Request....608

312. Start FCP Storage Discovery: Response....608

313. Get Connection Report: Request....613

314. Get Connection Report: Response (Part 1)....613

315. Get Connection Report: Response (Part 2)....614

316. Get Connection Report: Response (Part 3)....615

317. Get Connection Report: Response (Part 4)....616

318. Get Storage Group Histories: Request....627

319. Get Storage Group Histories: Response (Part 1)....628

320. Get Storage Group Histories: Response (Part 2)....629

321. Get Storage Group Histories: Response (Part 3)....630

322. Get Storage Group Histories: Response (Part 4)....631

323. Get Storage Group Histories: Response (Part 5)....632

324. Storage Group object: Sample inventory data - Response (Part 1)..... 634

325. Storage Group object: Sample inventory data - Response (Part 2)..... 635

326. List Storage Templates: Request.....643

327. List Storage Templates: Response..... 644

328. Create Storage Template: Request..... 647

329. Create Storage Template: Response.....647

330. Delete Storage Template: Request..... 648

331. Delete Storage Template: Response.....648

332. Get Storage Template Properties: Request..... 650

333. Get Storage Template Properties: Response.....650

334. Modify Storage Template Properties: Request..... 656

335. Modify Storage Template Properties: Response..... 656

336. List Storage Template Volumes of a Storage Template: Request..... 658

337. List Storage Template Volumes of a Storage Template: Response.....659

338. Get Storage Template Volume Properties: Request.....660

339. Get Storage Template Volume Properties: Response..... 660

340. Storage Template object: Sample inventory data - Response..... 661

341. List Tape Libraries: Request..... 665

342. List Tape Libraries: Response.....665

343. Undefine Tape Library: Request..... 666

344. Undefine Tape Library: Response..... 666

345. Get Tape Library Properties: Request..... 667

346. Get Tape Library Properties: Response.....668

347. Update Tape Library Properties: Request.....669

348. Update Tape Library Properties: Response..... 669

349. Request Tape Library Zoning: Request..... 672

350. Request Tape Library Zoning: Response..... 672

351. Discover Tape Libraries: Request..... 675

352. Discover Tape Libraries: Response..... 675

353. Tape Library object: Sample inventory data - Response..... 676

354. List Tape Links: Request..... 684

355. List Tape Links: Response..... 685

356. Create Tape Link: Request..... 689

357. Create Tape Link: Response..... 689

358. Get Tape Link Properties: Request..... 690

359. Get Tape Link Properties: Response..... 691

360. Modify Tape Link Properties: Request..... 694

361. Modify Tape Link Properties: Response..... 694

362. Delete Tape Link: Request..... 697

363. Delete Tape Link: Response..... 697

364. Add Adapter Ports: Request..... 699

365. Add Adapter Ports: Response..... 699

366. Remove Adapter Ports: Request..... 701

367. Remove Adapter Ports: Response..... 701

368. Replace Adapter Port: Request..... 704

369. Replace Adapter Port: Response..... 704

370. Resend Request: Request..... 706

371. Resend Request: Response..... 706

372. List Virtual Tape Resources of a Tape Link: Request..... 708

373. List Virtual Tape Resources of a Tape Link: Response..... 709

374. Get Virtual Tape Resource Properties: Request..... 710

375. Get Virtual Tape Resource Properties: Response..... 710

376. Update Virtual Tape Resource Properties: Request..... 713

377. Update Virtual Tape Resource Properties: Response..... 713

378. Get Partitions for a Tape Link: Request..... 715

379. Get Partitions for a Tape Link: Response..... 715

380. Get Tape Link Histories: Request..... 722

381. Get Tape Link Histories: Response (Part 1)..... 723

382. Get Tape Link Histories: Response (Part 2)..... 724

383. Get Tape Link Histories: Response (Part 3)..... 725

384. Get Tape Link Histories: Response (Part 4)..... 726

385. Update Tape Link Environment Report: Request..... 728

386. Update Tape Link Environment Report: Response (Part 1)..... 729

387. Get Tape Link Environment Report: Request..... 732

388. Get Tape Link Environment Report: Response (Part 1)..... 733

389. Get Tape Link Environment Report: Response (Part 2)..... 734

390. Tape Link object: Sample inventory data - Response..... 735

391. Create Partition Link: Request..... 755

392. Create Partition Link: Response..... 755

393. Delete Partition Link: Request..... 759

394. Delete Partition Link: Response..... 759

395. Get Partition Link Properties: Request..... 760

396. Get Partition Link Properties: Response..... 761

397. List Partition Links: Request..... 763

398. List Partition Links: Response..... 764

399. Modify Partition Link: Request..... 778

400. Modify Partition Link: Response..... 778

401. Partition Link object: Sample inventory data - Response..... 780

402. Get Console Properties: Request..... 813

403. Get Console Properties: Response (Part 1)..... 814

404. Get Console Properties: Response (Part 2)..... 815

405. Get Console Properties: Response (Part 3)..... 816

406. Get Console Properties: Response (Part 4)..... 817

407. Get Console Properties: Response (Part 5)..... 818

408. Get Console Properties: Response (Part 6)..... 819

409. Shutdown Console: Request..... 822

410. Shutdown Console: Response..... 822

411. Reorder User Patterns: Request..... 824

412. Reorder User Patterns: Response..... 824

413. Get Console Audit Log: Request..... 827

414. Get Console Audit Log: Response (Part 1)..... 828

415. Get Console Audit Log: Response (Part 2)..... 829

416. Get Console Audit Log: Response (Part 3)..... 830

417. Get Console Security Log: Request..... 832

418. Get Console Security Log: Response (Part 1)..... 832

419. Get Console Security Log: Response (Part 2)..... 833

420. Get Console Events Log: Request..... 835

421. Get Console Events Log: Response (Part 1)..... 836

422. Get Console Events Log: Response (Part 2)..... 837

423. Get Console Events Log: Response (Part 3)..... 838

424. List Console Hardware Messages: Request.....840

425. List Console Hardware Messages: Response.....841

426. Get Console Hardware Message Properties: Request.....842

427. Get Console Hardware Message Properties: Response.....843

428. Delete Console Hardware Message: Request.....844

429. Delete Console Hardware Message: Response.....844

430. Request Console Service: Request.....846

431. Request Console Service: Response.....846

432. Get Console Service Request Information: Request.....848

433. Get Console Service Request Information: Response.....848

434. Decline Console Service: Request.....849

435. Decline Console Service: Response.....849

436. List Unmanaged CPCs: Request.....851

437. List Unmanaged CPCs: Response.....851

438. Get Mobile App Preferences: Request.....852

439. Get Mobile App Preferences: Response.....853

440. Set Mobile App Preferences: Request.....856

441. Set Mobile App Preferences: Response.....856

442. Get CPC Notification Preferences for Device: Request.....859

443. Get CPC Notification Preferences for Device: Response.....860

444. Update CPC Notification Preferences for Device: Request.....863

445. Update CPC Notification Preferences for Device: Response.....864

446. List Remote Firmware Updates of the Console: Request.....866

447. List Remote Firmware Updates of the Console: Response.....866

448. Get Console Remote Firmware Update Properties: Request.....868

449. Get Console Remote Firmware Update Properties: Response..... 868

450. Delete Console Remote Firmware Update: Request..... 869

451. Delete Console Remote Firmware Update: Response..... 869

452. Authorize Remote Firmware Updates: Request..... 871

453. Authorize Remote Firmware Updates: Response..... 871

454. Update Welcome Text: Request..... 873

455. Update Welcome Text: Response..... 873

456. Get Console Notification Preferences for Device: Request..... 874

457. Get Console Notification Preferences for Device: Response..... 874

458. Update Console Notification Preferences for Device: Request..... 876

459. Update Console Notification Preferences for Device: Response..... 876

460. Console Single Step Install: Request..... 882

461. Console Single Step Install: Response..... 882

462. Report a Console Problem: Request..... 884

463. Report a Console Problem: Response..... 884

464. Console Delete Retrieved Internal Code: Request..... 887

465. Console Delete Retrieved Internal Code: Response..... 887

466. List Console API Features: Request..... 888

467. List Console API Features: Response..... 888

468. Console object: Sample inventory data (Part 1)..... 889

469. Console object: Sample inventory data (Part 2)..... 890

470. Console object: Sample inventory data (Part 3)..... 891

471. Console object: Sample inventory data (Part 4)..... 892

472. List Users: Request..... 902

473. List Users: Response..... 902

474. Get User Properties: Request....903

475. Get User Properties: Response....904

476. Update User Properties: Request....907

477. Update User Properties: Response....907

478. Add User Role to User: Request....909

479. Add User Role to User: Response....909

480. Remove User Role from User: Request....911

481. Remove User Role from User: Response....911

482. Create User: Request....915

483. Create User: Response....915

484. Delete User: Request....916

485. Delete User: Response....917

486. User object: Sample inventory data....918

487. List User Roles: Request....923

488. List User Roles: Response....923

489. Get User Role Properties: Request....924

490. Get User Role Properties: Response....925

491. Update User Role Properties: Request....927

492. Update User Role Properties: Response....927

493. Add Permission to User Role: Request....930

494. Add Permission to User Role: Response....930

495. Remove Permission from User Role: Request....933

496. Remove Permission from User Role: Response....933

497. Create User Role: Request....935

498. Create User Role: Response....935

499. Delete User Role: Request....936

500. Delete User Role: Response....936

501. User Role object: Sample inventory data....937

502. List Tasks: Request....939

503. List Tasks: Response....940

504. Get Task Properties: Request....941

505. Get Task Properties: Response....941

506. Task object: Sample inventory data....942

507. List User Patterns: Request....947

508. List User Patterns: Response....947

509. Get User Pattern Properties: Request....948

510. Get User Pattern Properties: Response....949

511. Update User Pattern Properties: Request....951

512. Update User Pattern Properties: Response....952

513. Create User Pattern: Request....955

514. Create User Pattern: Response....955

515. Delete User Pattern: Request....956

516. Delete User Pattern: Response....956

517. User Pattern object: Sample inventory data....957

518. List Password Rules: Request....962

519. List Password Rules: Response....963

520. Get Password Rule Properties: Request....964

521. Get Password Rule Properties: Response....965

522. Update Password Rule Properties: Request....966

523. Update Password Rule Properties: Response....967

524. Create Password Rule: Request..... 969

525. Create Password Rule: Response..... 969

526. Delete Password Rule: Request..... 970

527. Delete Password Rule: Response..... 970

528. Password Rule object: Sample inventory data..... 971

529. List LDAP Server Definitions: Request..... 978

530. List LDAP Server Definitions: Response..... 978

531. Get LDAP Server Definition Properties: Request..... 979

532. Get LDAP Server Definition Properties: Response..... 980

533. Update LDAP Server Definition Properties: Request..... 981

534. Update LDAP Server Definition Properties: Response..... 981

535. Create LDAP Server Definition: Request..... 984

536. Create LDAP Server Definition: Response..... 984

537. Delete LDAP Server Definition: Request..... 985

538. Delete LDAP Server Definition: Response..... 986

539. LDAP Server Definition object: Sample inventory data..... 986

540. List MFA Server Definitions: Request..... 989

541. List MFA Server Definitions: Response..... 989

542. Get MFA Server Definition Properties: Request..... 991

543. Get MFA Server Definition Properties: Response..... 991

544. Update MFA Server Definition Properties: Request..... 992

545. Update MFA Server Definition Properties: Response..... 992

546. Create MFA Server Definition: Request..... 994

547. Create MFA Server Definition: Response..... 994

548. Delete MFA Server Definition: Request..... 996

549. Delete MFA Server Definition: Response..... 996

550. MFA Server Definition object: Sample inventory data..... 996

551. List Custom Groups: Request.....1000

552. List Custom Groups: Response..... 1000

553. Get Custom Group Properties: Request..... 1001

554. Get Custom Group Properties: Response.....1002

555. Create Custom Group: Request..... 1003

556. Create Custom Group: Response.....1004

557. Delete Custom Group: Request.....1005

558. Delete Custom Group: Response.....1005

559. Add Member to Custom Group: Request.....1006

560. Add Member to Custom Group: Response..... 1007

561. Remove Member from Custom Group: Request.....1008

562. Remove Member from Custom Group: Response..... 1008

563. List Custom Group Members: Request..... 1010

564. List Custom Group Members: Response.....1010

565. List CPC Objects: Request.....1038

566. List CPC Objects: Response..... 1039

567. Get CPC Properties: Request..... 1041

568. Get CPC Properties: Response (Part 1).....1042

569. Get CPC Properties: Response (Part 2).....1043

570. Get CPC Properties: Response (Part 3).....1044

571. Get CPC Properties: Response (Part 4).....1045

572. Get CPC Properties: Response (Part 5).....1046

573. Get CPC Properties: Response (Part 6).....1047

574. Set Auto-Start List: Request....1064

575. Set Auto-Start List: Response....1064

576. Get CPC Audit Log: Request....1080

577. Get CPC Audit Log: Response....1081

578. Get CPC Security Log: Request....1083

579. Get CPC Security Log: Response....1084

580. Get CPC Events Log: Request....1086

581. Get CPC Events Log: Response (Part 1)....1087

582. Get CPC Events Log: Response (Part 2)....1088

583. List CPC Hardware Messages: Request....1090

584. List CPC Hardware Messages: Response....1091

585. Get CPC Hardware Message Properties: Request....1092

586. Get CPC Hardware Message Properties: Response....1093

587. Delete CPC Hardware Message: Request....1095

588. Delete CPC Hardware Message: Response....1095

589. Request CPC Service: Request....1097

590. Request CPC Service: Response....1097

591. Get CPC Service Request Information: Request....1099

592. Get CPC Service Request Information: Response....1099

593. Decline CPC Service: Request....1101

594. Decline CPC Service: Response....1101

595. Export WWPN List: Request....1103

596. Export WWPN List: Response....1103

597. Import DPM Configuration: Request (Part 1)....1110

598. Import DPM Configuration: Request (Part 2)....1111

599. Import DPM Configuration: Request (Part 3)....1112

600. Import DPM Configuration: Request (Part 4)....1113

601. Import DPM Configuration: Request (Part 5)....1114

602. Import DPM Configuration: Response....1114

603. List Remote Firmware Updates of a CPC: Request....1117

604. List Remote Firmware Updates of a CPC: Response....1117

605. Get CPC Remote Firmware Update Properties: Request....1119

606. Get CPC Remote Firmware Update Properties: Response....1119

607. Delete CPC Remote Firmware Update: Request....1121

608. Delete CPC Remote Firmware Update: Response....1121

609. Get LPAR Controls: Request....1132

610. Get LPAR Controls: Response (part 1)....1133

611. Get LPAR Controls: Response (part 2)....1134

612. Update LPAR Controls: Request....1136

613. Update LPAR Controls: Response....1136

614. CPC Single Step Install: Request....1141

615. CPC Single Step Install: Response....1141

616. Import Secure Execution Key: Request....1144

617. Import Secure Execution Key: Response....1144

618. Delete Secure Execution Key: Request....1146

619. Delete Secure Execution Key: Response....1146

620. Import CPC Certificate: Request....1149

621. Import CPC Certificate: Response....1149

622. Report a CPC Problem: Request....1151

623. Report a CPC Problem: Response....1151

624. Get CPC Historical Sustainability Data: Request..... 1155

625. Get CPC Historical Sustainability Data: Response..... 1156

626. CPC Install and Activate: Request..... 1162

627. CPC Install and Activate: Response..... 1162

628. CPC Delete Retrieved Internal Code: Request..... 1165

629. CPC Delete Retrieved Internal Code: Response..... 1165

630. List CPC API Features: Request..... 1166

631. List CPC API Features: Response..... 1166

632. Switch Support Elements: Request..... 1169

633. Switch Support Elements: Response..... 1169

634. List Logical Partitions of CPC: Request..... 1196

635. List Logical Partitions of CPC: Response..... 1196

636. List Permitted Logical Partitions: Request..... 1199

637. List Permitted Logical Partitions: Response..... 1200

638. Get Logical Partition Properties: Request..... 1202

639. Get Logical Partition Properties: Response (Part 1)..... 1203

640. Get Logical Partition Properties: Response (Part 2)..... 1204

641. Get Logical Partition Properties: Response (Part 3)..... 1205

642. Send OS Command: Request..... 1237

643. Send OS Command: Response..... 1237

644. Open OS Message Channel: Request..... 1239

645. Open OS Message Channel: Response..... 1239

646. List OS Messages of a Logical Partition: Request..... 1242

647. List OS Messages of a Logical Partition: Response..... 1243

648. Delete Logical Partition OS Message: Request..... 1245

649. Delete Logical Partition OS Message: Response..... 1245

650. Assign Certificate to Logical Partition: Request.....1258

651. Assign Certificate to Logical Partition: Response.....1258

652. Unassign Certificate from Logical Partition: Request.....1260

653. Unassign Certificate from Logical Partition: Response..... 1260

654. Report a Logical Partition Problem: Request.....1262

655. Report a Logical Partition Problem: Response..... 1262

656. Get Logical Partition Historical Sustainability Data: Request..... 1265

657. Get Logical Partition Historical Sustainability Data: Response.....1265

658. Delete Certificate: Request..... 1268

659. Delete Certificate: Response.....1268

660. Get Certificate Properties: Request..... 1270

661. Get Certificate Properties: Response.....1271

662. Get Encoded Certificate: Request.....1272

663. Get Encoded Certificate: Response..... 1272

664. List Certificates: Request..... 1275

665. List Certificates: Response.....1275

666. Update Certificate Properties: Request.....1277

667. Update Certificate Properties: Response..... 1277

668. User object: Sample inventory data.....1277

669. List Reset Activation Profiles: Request..... 1281

670. List Reset Activation Profiles: Response.....1281

671. Get Reset Activation Profile Properties: Request.....1283

672. Get Reset Activation Profile Properties: Response..... 1283

673. Create Reset Activation Profile: Request.....1289

674. Create Reset Activation Profile: Response..... 1289

675. Delete Reset Activation Profile: Request..... 1290

676. Delete Reset Activation Profile: Response..... 1291

677. List Image Activation Profiles: Request..... 1309

678. List Image Activation Profiles: Response..... 1310

679. Get Image Activation Profile Properties: Request..... 1312

680. Get Image Activation Profile Properties: Response (Part 1)..... 1313

681. Get Image Activation Profile Properties: Response (Part 2)..... 1314

682. Assign Certificate to Image Activation Profile: Request..... 1318

683. Assign Certificate to Image Activation Profile: Response..... 1319

684. Unassign Certificate from Image Activation Profile: Request..... 1321

685. Unassign Certificate from Image Activation Profile: Response..... 1321

686. Create Image Activation Profile: Request..... 1338

687. Create Image Activation Profile: Response..... 1338

688. Delete Image Activation Profile: Request..... 1339

689. Delete Image Activation Profile: Response..... 1340

690. List Load Activation Profiles: Request..... 1348

691. List Load Activation Profiles: Response..... 1348

692. Get Load Activation Profile Properties: Request..... 1350

693. Get Load Activation Profile Properties: Response..... 1351

694. Create Load Activation Profile: Request..... 1360

695. Create Load Activation Profile: Response..... 1360

696. Delete Load Activation Profile: Request..... 1361

697. Delete Load Activation Profile: Response..... 1362

698. List Group Profiles: Request..... 1366

699. List Group Profiles: Response..... 1366

700. Get Group Profile Properties: Request..... 1368

701. Get Group Profile Properties: Response..... 1368

702. Create Group Profile: Request..... 1373

703. Create Group Profile: Response..... 1373

704. Delete Group Profile: Request..... 1375

705. Delete Group Profile: Response..... 1375

706. Energy management as applied throughout layers of enterprise management..... 1383

707. Example of a CPC group that contains a zCPC..... 1385

708. Get Energy Optimization Advice Summary: Request..... 1401

709. Get Energy Optimization Advice Summary: Response..... 1402

710. Get Energy Optimization Advice Details: Request..... 1406

711. Get Energy Optimization Advice Details: Response (Part 1)..... 1407

712. Get Energy Optimization Advice Details: Response (Part 2)..... 1408

713. Get Energy Optimization Advice Details: Response (Part 3)..... 1409

714. Get Energy Optimization Advice Details: Response (Part 4)..... 1410

Tables

1. Summary of updates for API version 1.1 (HMC/SE Version 2.11.1)..... 8

2. Summary of updates for API version 1.2 (HMC/SE Version 2.11.1)..... 8

3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0)..... 9

4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1)..... 10

5. Summary of updates for API version 1.5 (HMC/SE Version 2.12.1)..... 12

6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0)..... 13

7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1)..... 15

8. Summary of updates for API version 2.1 (HMC/SE Version 2.13.1)..... 19

9. Summary of updates for API version 2.2 (HMC/SE Version 2.13.1)..... 20

10. Summary of updates for API version 2.3 (HMC/SE Version 2.13.1)..... 20

11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0)..... 21

12. Summary of updates for API version 2.21 (HMC/SE Version 2.14.0)..... 24

13. Summary of updates for API version 2.22 (HMC/SE Version 2.14.0)..... 24

14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0)..... 24

15. Summary of updates for API version 2.24 (HMC/SE Version 2.14.0)..... 26

16. Summary of updates for API version 2.25 (HMC/SE Version 2.14.0)..... 27

17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1)..... 28

18. Summary of updates for API version 2.36 (HMC/SE Version 2.14.1)..... 29

19. Summary of updates for API version 2.37 (HMC/SE Version 2.14.1)..... 30

20. Summary of updates for API version 2.38 (HMC/SE Version 2.14.1)..... 30

21. Summary of updates for API version 2.39 (HMC/SE Version 2.14.1)..... 30

22. Summary of updates for API version 2.40 (HMC/SE Version 2.14.1)..... 30

23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0)..... 31

24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0)....35

25. Summary of updates for API version 3.3 (HMC/SE Version 2.15.0)....36

26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0)....37

27. Summary of updates for API version 3.5 (HMC/SE Version 2.15.0)....38

28. Summary of updates for API version 3.6 (HMC/SE Version 2.15.0)....38

29. Summary of updates for API version 3.7 (HMC/SE Version 2.15.0)....39

30. Summary of updates for API version 3.8 (HMC/SE Version 2.15.0)....40

31. Summary of updates for API version 3.9 (HMC/SE Version 2.15.0)....41

32. Summary of updates for API version 3.10 (HMC/SE Version 2.15.0).... 44

33. Summary of updates for API version 3.11 (HMC/SE Version 2.15.0).... 44

34. Summary of updates for API version 3.12 (HMC/SE Version 2.15.0).... 45

35. Summary of updates for API version 3.13 (HMC/SE Version 2.15.0).... 45

36. Summary of updates for API version 4.1 (HMC/SE Version 2.16.0)....46

37. Summary of updates for API version 4.2 (HMC/SE Version 2.16.0)....50

38. Summary of updates for API version 4.10 (HMC/SE Version 2.16.0).... 51

39. Summary of features for API version 4.10 (HMC/SE Version 2.16.0).... 52

40. Primitive data types.... 55

41. Compound data types....56

42. Primitive data types in JSON notation....57

43. Compound data types in JSON notation.... 58

44. error-feature-info object properties....72

45. SSE notification event properties.... 91

46. SSE common event properties.... 91

47. API Features.... 103

48. General API services: operations summary....111

49. General API services: URI variables....112

50. Logon: HTTP status and reason codes....118

51. mfa-info-request nested object properties....119

52. change-password-info nested object properties....119

53. Provide Requested MFA Information: HTTP status and reason codes....124

54. mfa-requested-information-failures object....125

55. mfa-requested-information-failing-factor object....125

56. Change Logon Password: HTTP status and reason codes....128

57. Verify Logon Password: HTTP status and reason codes....129

58. topic-info object....132

59. object-change-filter nested object....136

60. property-change-filter nested object....137

61. os-message-filter nested object....137

62. Submit Requests: HTTP status and reason codes....149

63. Inventory service: operations summary....159

64. Metrics service: operations summary....159

65. Metrics service: URI variables....159

66. Channels metric group....177

67. CPC overview metric group....178

68. DPM system overview metric group....180

69. Logical partitions metric group....180

70. Partitions metric group....181

71. zCPC environmentals and power metric group....182

72. Power status metric group....183

73. zCPC processors metric group....186

74. Crypto metric group.....186

75. Adapters metric group.....187

76. Flash memory adapters metric group.....187

77. RoCE adapters metric group.....188

78. Network adapter port metric group.....188

79. Network interface metric group.....190

80. DPM - Partition: operations summary.....197

81. DPM - Partition: URI variables.....199

82. DPM - Adapter: operations summary.....200

83. DPM - Adapter: URI variables.....201

84. DPM - Virtual Switch: operations summary.....201

85. DPM - Virtual Switch: URI variables.....201

86. DPM - Capacity Group: operations summary.....201

87. DPM - Capacity Group: URI variables.....202

88. DPM - Storage Site: operations summary.....202

89. DPM - Storage Site: URI variables.....202

90. DPM - Storage Fabric: operations summary.....202

91. DPM - Storage Fabric: URI variables.....203

92. DPM - Storage Switch: operations summary.....203

93. DPM - Storage Switch: URI variables.....203

94. DPM - Storage Subsystem: operations summary.....204

95. DPM - Storage Subsystem: URI variables.....204

96. DPM - Storage Control Unit: operations summary.....204

97. DPM - Storage Control Unit: URI variables.....205

98. DPM - Storage Group: operations summary.....205

99. DPM - Storage Group: URI variables..... 206

100. DPM - Storage Template: operations summary..... 206

101. DPM - Storage Template: URI variables..... 207

102. DPM - Tape Library: operations summary..... 207

103. DPM - Tape Library: URI variables..... 207

104. DPM - Tape Link: operations summary..... 208

105. DPM - Tape Link: URI variables..... 208

106. DPM - Partition Link: operations summary..... 209

107. DPM - Partition Link: URI variables..... 209

108. Partition object: base managed object properties specializations..... 209

109. Partition object: class specific properties..... 211

110. partition-feature-info object properties..... 228

111. boot-record-location object properties [Added by feature secure-boot-with-certificates]..... 228

112. crypto-configuration nested object properties..... 229

113. crypto-domain-configuration nested object properties..... 229

114. Partition object - Virtual Function element properties..... 230

115. Partition object - NIC element object properties..... 230

116. Partition object - HBA element object properties..... 233

117. Create Partition: HTTP status and reason codes..... 244

118. Delete Partition: HTTP status and reason codes..... 246

119. Delete Partition Asynchronously: HTTP status and reason codes..... 248

120. Delete Partition Asynchronously: Job status and reason codes..... 249

121. Update Partition Properties: HTTP status and reason codes..... 255

122. Update Partition Properties Asynchronously: HTTP status and reason codes..... 259

123. Update Partition Properties Asynchronously: Job status and reason codes..... 259

124. Start Partition: Job status and reason codes....264

125. Attach Storage Group to Partition: HTTP status and reason codes....268

126. Stop Partition: Job status and reason codes....270

127. Dump Partition: HTTP status and reason codes....273

128. Dump Partition: Job status and reason codes....274

129. Start Dump Program: Job status and reason codes....280

130. Perform PSW Restart: Job status and reason codes....283

131. Create Virtual Function: HTTP status and reason codes....285

132. Delete Virtual Function: HTTP status and reason codes....287

133. Get Virtual Function Properties: HTTP status and reason codes....288

134. Update Virtual Function Properties: HTTP status and reason codes....290

135. Create NIC: HTTP status and reason codes....294

136. Delete NIC: HTTP status and reason codes....297

137. Get NIC Properties: HTTP status and reason codes....299

138. Update NIC Properties: HTTP status and reason codes....302

139. Increase Crypto Configuration: HTTP status and reason codes....306

140. Change Crypto Domain Configuration: HTTP status and reason codes....309

141. Decrease Crypto Configuration: HTTP status and reason codes....311

142. Zeroize Crypto Domain: HTTP status and reason codes....314

143. Mount ISO Image: HTTP status and reason codes....317

144. Unmount ISO Image: HTTP status and reason codes....318

145. Detach Storage Group from Partition: HTTP status and reason codes....320

146. Create HBA: HTTP status and reason codes....323

147. Delete HBA: HTTP status and reason codes....325

148. Update HBA Properties: HTTP status and reason codes....327

149. Get HBA Properties: HTTP status and reason codes....329

150. Reassign Storage Adapter Port: HTTP status and reason codes....331

151. Open OS Message Channel: HTTP status and reason codes....335

152. Attach Tape Link to Partition: HTTP status and reason codes.... 344

153. Detach Tape Link from Partition: HTTP status and reason codes....347

154. Assign Certificate to Partition: HTTP status and reason codes....354

155. Unassign Certificate from Partition: HTTP status and reason codes.... 356

156. Adapter object: base managed object properties specializations....360

157. Adapter object: class-specific properties....362

158. network-port-info object properties [Added by feature adapter-network-information]....372

159. Network Port element object properties.... 372

160. Storage Port element object properties....373

161. List Adapters of a CPC: HTTP status and reason codes.... 376

162. List Permitted Adapters: HTTP status and reason codes....380

163. Update Adapter Properties: HTTP status and reason codes....384

164. Change Crypto Type: HTTP status and reason codes....387

165. Create Hipersocket: HTTP status and reason codes....389

166. Delete Hipersocket: HTTP status and reason codes.... 391

167. Get Partitions Assigned to Adapter: HTTP status and reason codes....394

168. Get Network Port Properties: HTTP status and reason codes.... 395

169. Update Network Port Properties: HTTP status and reason codes....397

170. Get Storage Port Properties: HTTP status and reason codes....398

171. Update Storage Port Properties: HTTP status and reason codes.... 400

172. Change Adapter Type: HTTP status and reason codes.... 402

173. Update Adapter Firmware: HTTP status and reason codes.... 405

174. Update Adapter Firmware: HTTP status and reason codes..... 406

175. Virtual Switch object: base managed object properties specializations..... 408

176. Virtual Switch object: class specific properties..... 409

177. Update Virtual Switch Properties: HTTP status and reason codes..... 418

178. Capacity Group element object properties..... 420

179. Create Capacity Group: HTTP status and reason codes..... 425

180. Delete Capacity Group: HTTP status and reason codes..... 427

181. Get Capacity Group Properties: HTTP status and reason codes..... 428

182. Add Partition to Capacity Group: HTTP status and reason codes..... 430

183. Remove Partition from Capacity Group: HTTP status and reason codes..... 432

184. Update Capacity Group Properties: HTTP status and reason codes..... 434

185. Storage Site object properties..... 436

186. Storage Site object: class specific properties..... 436

187. List Storage Sites: HTTP status and reason codes..... 438

188. Create Storage Site: HTTP status and reason codes..... 440

189. Delete Storage Site: HTTP status and reason codes..... 443

190. Get Storage Site Properties: HTTP status and reason codes..... 444

191. Update Storage Site Properties: HTTP status and reason codes..... 446

192. Storage Fabric object properties..... 448

193. Storage Fabric object: class specific properties..... 449

194. List Storage Fabrics: HTTP status and reason codes..... 451

195. Create Storage Fabric: HTTP status and reason codes..... 453

196. Delete Storage Fabric: HTTP status and reason codes..... 455

197. Get Storage Fabric Properties: HTTP status and reason codes..... 456

198. Update Storage Fabric Properties: HTTP status and reason codes..... 458

199. Storage Switch object: base managed object properties specializations....460

200. Storage Switch object: class specific properties.... 461

201. List Storage Switches of a Storage Site: HTTP status and reason codes....463

202. List Storage Switches of a Storage Fabric: HTTP status and reason codes.... 465

203. Define Storage Switch: HTTP status and reason codes.... 467

204. Undefine Storage Switch: HTTP status and reason codes.... 469

205. Get Storage Switch Properties: HTTP status and reason codes.... 471

206. Update Storage Switch Properties: HTTP status and reason codes.... 472

207. Move Storage Switch to Storage Site: HTTP status and reason codes.... 474

208. Move Storage Switch to Storage Fabric: HTTP status and reason codes....476

209. Storage Subsystem object: base managed object properties specializations.... 478

210. Storage Subsystem object: class specific properties.... 479

211. subsystem-connection-endpoint nested object properties....480

212. List Storage Subsystems of a Storage Site: HTTP status and reason codes....481

213. Define Storage Subsystem: HTTP status and reason codes.... 483

214. Undefine Storage Subsystem: HTTP status and reason codes.... 485

215. Get Storage Subsystem Properties: HTTP status and reason codes.... 487

216. Update Storage Subsystem Properties: HTTP status and reason codes.... 489

217. Move Storage Subsystem to Storage Site: HTTP status and reason codes.... 490

218. Add Connection Endpoint: HTTP status and reason codes....493

219. Remove Connection Endpoint: HTTP status and reason codes.... 495

220. Storage Control Unit object: base managed object properties specializations....497

221. Storage Control Unit object: class specific properties.... 498

222. Storage Control Unit object: volume-range nested object properties.... 499

223. Storage Control Unit object: storage path element object properties.... 499

224. List Storage Control Units of a Storage Subsystem: HTTP status and reason codes..... 501

225. Define Storage Control Unit: HTTP status and reason codes.....503

226. Undefine Storage Control Unit: HTTP status and reason codes.....505

227. Get Storage Control Unit Properties: HTTP status and reason codes.....506

228. Update Storage Control Unit Properties: HTTP status and reason codes..... 508

229. Add Volume Range: HTTP status and reason codes.....510

230. Remove Volume Range: HTTP status and reason codes.....512

231. Create Storage Path: HTTP status and reason codes.....515

232. Delete Storage Path: HTTP status and reason codes.....517

233. Get Storage Path Properties: HTTP status and reason codes.....519

234. Update Storage Path Properties: HTTP status and reason codes.....521

235. Storage Group object: base managed object properties specializations..... 526

236. Storage Group object: class specific properties..... 526

237. Storage Volume element object properties..... 532

238. partition-volume-path-info nested object..... 542

239. Virtual Storage Resource element object properties..... 542

240. world-wide-port-name-info object: properties..... 544

241. List Storage Groups: HTTP status and reason codes..... 546

242. Create Storage Group: HTTP status and reason codes..... 551

243. created-object-info object.....552

244. Delete Storage Group: HTTP status and reason codes..... 555

245. Get Storage Group Properties: HTTP status and reason codes..... 557

246. storage-volume-request-info nested object for "create" operations on "fc" storage volumes..... 560

247. storage-volume-request-info nested object for "create" operations on "fcp" storage volumes..... 561

248. storage-volume-request-info nested object for "create" operations on "nvme" storage volumes... 561

249. storage-volume-request-info nested object for "modify" operations on "fc" storage volumes..... 562

250. storage-volume-request-info nested object for "modify" operations on "fcp" storage volumes..... 563

251. storage-volume-request-info nested object for "modify" operations on "nvme" storage volumes.. 563

252. storage-volume-request-info nested object for "delete" operations on storage volumes of all types..... 564

253. Modify Storage Group Properties: HTTP status and reason codes..... 566

254. Resend Request: HTTP status and reason codes..... 570

255. Add Candidate Adapter Ports to an FCP Storage Group: HTTP status and reason codes.....572

256. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes.....574

257. List Storage Volumes of a Storage Group: HTTP status and reason codes.....577

258. Get Storage Volume Properties: HTTP status and reason codes..... 579

259. Fulfill FICON Storage Volume: HTTP status and reason codes..... 582

260. Fulfill FICON Storage Volumes: HTTP status and reason codes..... 585

261. Fulfill FCP Storage Volume: HTTP status and reason codes..... 588

262. Accept Mismatched Storage Volumes: HTTP status and reason codes..... 590

263. Reject Mismatched FCP Storage Volumes: HTTP status and reason codes..... 593

264. List Virtual Storage Resources of a Storage Group: HTTP status and reason codes..... 596

265. Get Virtual Storage Resource Properties: HTTP status and reason codes..... 597

266. Update Virtual Storage Resource Properties: HTTP status and reason codes..... 599

267. Get Partitions for a Storage Group: HTTP status and reason codes..... 602

268. Validate LUN Path: HTTP status and reason codes..... 604

269. Start FCP Storage Discovery: HTTP status and reason codes.....607

270. Start FCP Storage Discovery: Job status and reason codes.....607

271. fcp-fabric-info nested object.....609

272. world-wide-port-name-zone-info nested object.....610

273. adapter-info nested object..... 610

274. fcp-storage-subsystem-info nested object..... 611

275. storage-configuration-info nested object..... 611

276. ficon-storage-subsystem-info nested object..... 612

277. Get Connection Report: HTTP status and reason codes..... 612

278. storage-group-history-info nested object..... 618

279. storage-group-action-info nested object..... 618

280. storage-group-configuration nested object..... 621

281. storage-volume-info nested object..... 623

282. virtual-storage-resource-info nested object..... 625

283. Get Storage Group Histories: HTTP status and reason codes..... 627

284. Storage Template object: base managed object properties specializations..... 635

285. Storage Template object: class specific properties..... 636

286. Storage Template Volume element object properties..... 638

287. List Storage Templates: HTTP status and reason codes..... 643

288. Create Storage Template: HTTP status and reason codes..... 646

289. Delete Storage Template: HTTP status and reason codes..... 648

290. Get Storage Template Properties: HTTP status and reason codes..... 649

291. storage-template-volume-request-info nested object for "create" operations on "fc" storage template volumes..... 651

292. storage-template-volume-request-info nested object for "create" operations on "fcp" storage template volumes..... 652

293. storage-template-volume-request-info nested object for "modify" operations on "fc" storage template volumes..... 652

294. storage-template-volume-request-info nested object for "modify" operations on "fcp" storage template volumes..... 653

295. storage-template-volume-request-info nested object for "delete" operations on "fc" or "fcp" storage template volumes..... 653

296. Modify Storage Template Properties: HTTP status and reason codes..... 655

297. List Storage Template Volumes of a Storage Template: HTTP status and reason codes.....658

298. Get Storage Template Volume Properties: HTTP status and reason codes.....660

299. Tape Library object: base managed object properties specializations.....662

300. Tape Library object: class specific properties.....662

301. List Tape Libraries: HTTP status and reason codes.....664

302. Undefine Tape Library: HTTP status and reason codes.....666

303. Get Tape Library Properties: HTTP status and reason codes.....667

304. Update Tape Library Properties: HTTP status and reason codes.....669

305. Request Tape Library Zoning: HTTP status and reason codes.....671

306. Discover Tape Libraries: HTTP status and reason codes.....674

307. Discover Tape Libraries: Job status and reason codes.....675

308. 676

309. Tape Link object: base managed object properties specializations.....677

310. Tape Link object: class specific properties.....678

311. Tape Link object - Virtual Tape Resource element object properties.....681

312. List Tape Links: HTTP status and reason codes.....684

313. Create Tape Link: HTTP status and reason codes.....687

314. Get Tape Link Properties: HTTP status and reason codes.....690

315. Modify Tape Link Properties: HTTP status and reason codes.....693

316. Delete Tape Link: HTTP status and reason codes.....696

317. Add Adapter Ports: HTTP status and reason codes.....698

318. Remove Adapter Ports: HTTP status and reason codes.....700

319. Replace Adapter Port: HTTP status and reason codes.....703

320. Resend Request: HTTP status and reason codes.....706

321. List Virtual Tape Resources of a Tape Link: HTTP status and reason codes.....708

322. Get Virtual Tape Resource Properties: HTTP status and reason codes.....710

323. Update Virtual Tape Resource Properties: HTTP status and reason codes.....712

324. Get Partitions for a Tape Link: HTTP status and reason codes.....714

325. tape-link-history-info nested object.....716

326. tape-link-action-info nested object.....716

327. tape-link-configuration nested object.....718

328. virtual-tape-resource-info nested object.....721

329. Get Tape Link Histories: HTTP status and reason codes.....722

330. 727

331. Update Tape Link Environment Report: HTTP status and reason codes.....728

332. Update Tape Link Environment Report: Job status and reason codes.....728

333. fabric-info nested object.....730

334. world-wide-port-name-zone-info nested object.....730

335. adapter-info nested object.....730

336. tape-library-info nested object.....731

337. Get Tape Link Environment Report: HTTP status and reason codes.....732

338. Partition Link object: base managed object properties specializations.....736

339. Partition Link object: class specific properties.....737

340. pending-operations nested object properties.....739

341. ctc-path-details nested object properties.....739

342. ctc-endpoints-details nested object properties.....739

343. adapter-info nested object properties.....740

344. ctc-partition-device-endpoint-details nested object properties.....740

345. bus-connection nested object properties.....740

346. nic nested object properties.....741

347. new-bus-connection nested object properties.... 743
348. new-nic nested object properties.... 744
349. added-ctc-path-info nested object properties.... 745
350. ctc-endpoint nested object properties.... 746
351. ctc-partition-devices-endpoint nested object properties.... 746
352. Table 9. error-job-results nested object.... 747
353. Create Partition Link: HTTP status and reason codes.... 750
354. conflicting-device-numbers nested object.... 753
355. conflicting-device-info nested object.... 753
356. invalid-fid-details nested object.... 754
357. invalid-path-error nested object.... 754
358. adapter-uris nested object.... 754
359. invalid-mac-details nested object.... 754
360. not-found-details nested object.... 754
361. Create Partition Link: Job status and reason codes.... 754
362. job-accepted-response nested object.... 756
363. Table error-job-results nested object.... 756
364. Delete Partition Link: HTTP status and reason codes.... 757
365. Get Partition Link Properties: HTTP status and reason codes.... 760
366. List Partition Links: HTTP status and reason codes.... 763
367. modified-bus-connection nested object.... 766
368. modified-nic nested object.... 766
369. removed-ctc-path-info nested object.... 767
370. modified-ctc-path-info nested object.... 767
371. job-accepted-response nested object.... 768

372. error-job-results nested object....769

373. Modify Partition Link: HTTP status and reason codes....772

374. nic-not-found-details nested object....777

375. partitions-exist-info nested object....777

376. partitions-exist-details nested object....777

377. Modify Partition Link: Job status and reason codes....777

378. Core IBM zSystems resources - Console: operations summary....781

379. Core IBM zSystems resources - Console: URI variables....783

380. Core IBM zSystems resources - User: operations summary....783

381. Core IBM zSystems resources - User: URI variables....784

382. Core IBM zSystems resources - User Role: operations summary....784

383. Core IBM zSystems resources - User Role: URI variables....784

384. Core IBM zSystems resources - Task: operations summary....785

385. Core IBM zSystems resources - Task: URI variables....785

386. Core IBM zSystems resources - User Pattern: operations summary....785

387. Core IBM zSystems resources - User Pattern: URI variables....785

388. Core IBM zSystems resources - Password Rule: operations summary....785

389. Core IBM zSystems resources - Password Rule: URI variables....786

390. Core IBM zSystems resources - LDAP Server Definition: operations summary....786

391. Core IBM zSystems resources - LDAP Server Definition: URI variables....786

392. Core IBM zSystems resources - MFA Server Definition: operations summary....787

393. Core IBM zSystems resources - MFA Server Definition: URI variables....787

394. Core IBM zSystems resources - Group: operations summary....787

395. Core IBM zSystems resources - Groups: URI variables....788

396. Core IBM zSystems resources - CPC: operations summary....788

397. Core IBM zSystems resources - CPC: URI variables....790

398. Core IBM zSystems resources - Logical partitions: operations summary.... 791

399. Core IBM zSystems resources - Logical partitions: URI variables.... 792

400. Core IBM zSystems resources - Certificate: operations summary.... 792

401. Core IBM zSystems resources - Certificate: URI variables.... 793

402. Core IBM zSystems resources - Reset activation profile: operations summary....793

403. Core IBM zSystems resources - Image activation profile: operations summary.... 793

404. Core IBM zSystems resources - Load activation profile: operations summary....794

405. Core IBM zSystems resources - Group profile: operations summary....794

406. Core IBM zSystems resources - Activation profile: URI variables.... 795

407. Core IBM zSystems resources - Capacity record: operations summary....795

408. Core IBM zSystems resources - Capacity record: URI variables.... 795

409. Core IBM zSystems resources - Adapter: operations summary.... 795

410. ec-mcl-description object.... 796

411. action object.... 797

412. ec object....797

413. mcl object.... 798

414. stp-config object....798

415. stp-node object....799

416. psw-description object....799

417. zaware-network object....799

418. ssc-network object.... 800

419. ip-info object....800

420. network-ip-info object....800

421. absolute-capping object....801

422. Console object: base managed object properties specializations....801

423. Console object: class specific additional properties....802

424. network-info object properties....803

425. detailed-network-info properties....804

426. ipv4-info properties....804

427. ipv6-info properties....804

428. machine-info properties....805

429. hardware-message object properties....805

430. hardware-message-details base properties....806

431. hardware-message-details type-specific properties when the type value is "basic":....806

432. hardware-message-details type-specific properties when the type value is "common-problem":..806

433. hardware-message-node-details:....806

434. hardware-message-data-details:....807

435. mobile-app-preferences object properties....807

436. action-settings object properties....808

437. hma-info object properties....809

438. hma-guest-info object properties....809

439. hma-guest-info object type-specific properties when the type value is "se"....809

440. Console object - Remote Firmware Update element object properties....810

441. Console object - Remote Firmware Update Execution Step Console nested object properties....812

442. Reorder User Patterns: HTTP status and reason codes....823

443. log-entry-info object properties....825

444. event-details-info object properties....825

445. event-data-item-info object properties....826

446. List Console Hardware Messages: HTTP status and reason codes....840

447. List Unmanaged CPCs: HTTP status and reason codes....851

448. List Remote Firmware Updates of the Console: remote-firmware-update-info objects....864

449. List Remote Firmware Updates of the Console: remote-firmware-update-token-info objects....865

450. List Remote Firmware Updates of the Console: HTTP status and reason codes.... 866

451. Get Console Remote Firmware Update Properties: HTTP status and reason codes....867

452. Delete Console Remote Firmware Update: HTTP status and reason codes....869

453. Authorize Remote Firmware Updates: HTTP status and reason codes....871

454. Get Console Notification Preferences for Device: HTTP status and reason codes....874

455. Update Console Notification Preferences for Device: HTTP status and reason codes.... 875

456. User object: base managed object properties specializations....893

457. User object: class specific additional properties....894

458. List Users: HTTP status and reason codes....901

459. Update User Properties: HTTP status and reason codes.... 906

460. Add User Role to User: HTTP status and reason codes....908

461. Remove User Role from User: HTTP status and reason codes.... 910

462. Create User: HTTP status and reason codes.... 914

463. Delete User: HTTP status and reason codes.... 916

464. User Role object: base managed object properties specializations.... 919

465. User Role object: class specific additional properties....919

466. permission-info object properties....920

467. Update User Role Properties: HTTP status and reason codes....926

468. Add Permission to User Role: HTTP status and reason codes....929

469. Remove Permission from User Role: HTTP status and reason codes....932

470. Create User Role: HTTP status and reason codes.... 934

471. Delete User Role: HTTP status and reason codes.... 936

472. Task object: properties..... 938

473. User Pattern object: properties..... 943

474. group-to-template-mapping properties..... 945

475. Update User Pattern Properties: HTTP status and reason codes..... 950

476. Create User Pattern: HTTP status and reason codes..... 954

477. Password Rule object: properties..... 958

478. character-rule object properties..... 960

479. custom-character-set object properties..... 960

480. Update Password Rule Properties: HTTP status and reason codes..... 966

481. Create Password Rule: HTTP status and reason codes..... 968

482. LDAP Server Definition object: properties..... 972

483. Create LDAP Server Definition: HTTP status and reason codes..... 984

484. MFA Server Definition object: properties..... 987

485. Create MFA Server Definition: HTTP status and reason codes..... 994

486. Group object: base managed object properties specializations..... 997

487. Group object: class specific additional properties..... 998

488. match-info object properties..... 998

489. CPC object: base managed object properties specializations..... 1011

490. CPC object: class specific additional properties..... 1014

491. ipv6-info object properties..... 1024

492. hardware-message object properties..... 1025

493. cpc-feature-info object properties..... 1025

494. auto-start-entry object base properties..... 1026

495. auto-start-entry object type-specific properties when type value is "partition"..... 1026

496. auto-start-entry object type-specific properties when type value is "partition-group"..... 1026

497. CPC object: energy management related additional properties....1027

498. CPC object - Remote Firmware Update element object properties....1033

499. CPC object - Remote Firmware Update Execution Step CPC nested object properties....1036

500. Set Auto-Start List: HTTP status and reason codes.... 1063

501. Get CPC Audit Log: HTTP status and reason codes....1080

502. Get CPC Security Log: HTTP status and reason codes....1083

503. List CPC Hardware Messages: HTTP status and reason codes....1090

504. Get CPC Hardware Message Properties: HTTP status and reason codes....1092

505. Delete CPC Hardware Message: HTTP status and reason codes....1094

506. Export WWPN List: HTTP status and reason codes....1103

507. cpc-info nested object 1107

508. adapter-mapping-info nested object 1107

509. List Remote Firmware Updates of a CPC: remote-firmware-update-info objects.... 1116

510. List Remote Firmware Updates of a CPC: HTTP status and reason codes.... 1116

511. Get CPC Remote Firmware Update Properties: HTTP status and reason codes....1118

512. Delete CPC Remote Firmware Update: HTTP status and reason codes....1120

513. Get Logical Partition Resource Assignments: HTTP status and reason codes....1122

514. Update LPAR Controls: HTTP status and reason codes.... 1135

515. 1144

516. 1144

517. integer-data-point nested object....1153

518. float-data-point nested object....1154

519. ec-level nested object....1157

520. Logical Partition object: base managed object properties specializations.... 1170

521. Logical Partition object: class specific additional properties....1171

522. boost-info nested object properties..... 1193

523. central-storage-allocation nested object properties..... 1193

524. expanded-storage-allocation nested object properties..... 1194

525. Open OS Message Channel: HTTP status and reason codes..... 1238

526. 1256

527. Certificate object: base managed object properties specializations..... 1266

528. Certificate object: class specific additional properties..... 1267

529. Reset activation profile: properties..... 1278

530. fenced-book-data nested object properties..... 1287

531. 1289

532. Image activation profile: properties..... 1291

533. assigned-crypto-domain nested object..... 1307

534. assigned-crypto nested object..... 1307

535. 1317

536. 1338

537. Load activation profile: properties..... 1340

538. 1360

539. Group profile: properties..... 1363

540. 1373

541. Capacity records: class-specific properties..... 1376

542. caprec-proc-info object..... 1378

543. caprec-target object..... 1378

544. Energy management: operations summary..... 1387

545. Energy management: URI variables..... 1387

546. Get Energy Optimization Advice Summary: HTTP status and reason codes..... 1401

547. Get Energy Optimization Advice Details: HTTP status and reason codes..... 1406

548. disabled-wait-filter nested object.....1431

549. Enum values for a class of managed objects.....1437

550. Enum values for the name property of User Role objects with a type of "system-defined" 1439

551. Enum values for the name property of Task objects..... 1441

Safety

Safety notices

Safety notices may be printed throughout this guide. **DANGER** notices warn you of conditions or procedures that can result in death or severe personal injury. **CAUTION** notices warn you of conditions or procedures that can cause personal injury that is neither lethal nor extremely hazardous. **Attention** notices warn you of conditions or procedures that can cause damage to machines, equipment, or programs.

World trade safety information

Several countries require the safety information contained in product publications to be provided in their local language(s). If this requirement applies to your country, a safety information booklet is included in the publications package shipped with the product. The booklet contains the translated safety information with references to the US English source. Before using a US English publication to install, operate, or service this product, you must first become familiar with the related safety information in the *Systems Safety Notices*, G229-9054. You should also refer to the booklet any time you do not clearly understand any safety information in the US English publications.

Laser safety information

All IBM Z and IBM LinuxONE (LinuxONE) models can use I/O cards such as FICON®, Open Systems Adapter (OSA), RoCE Express, Integrated Coupling Adapter (ICA SR, ICA SR1.1), zHyperLink Express, or other I/O features which are fiber optic based and utilize lasers (short wavelength or long wavelength lasers).

Laser compliance

All lasers are certified in the US to conform to the requirements of DHHS 21 CFR Subchapter J for Class 1 or Class 1M laser products. Outside the US, they are certified to be in compliance with IEC 60825 as a Class 1 or Class 1M laser product. Consult the label on each part for laser certification numbers and approval information.

Laser Notice: U.S. FDA CDRH NOTICE if low power lasers are utilized, integrated, or offered with end product systems as applicable. Complies with 21 CFR 1040.10 and 1040.11 except for conformance with IEC 60825-1 Ed. 3., as described in Laser Notice No. 56, dated May 8, 2019.

CAUTION: This product might contain one or more of the following devices: CD-ROM drive, DVDROM drive, DVD-RAM drive, or laser module, which are Class 1 laser products. Note the following information:

- **Do not remove the covers. Removing the covers of the laser product could result in exposure to hazardous laser radiation. There are no serviceable parts inside the device.**
- **Use of the controls or adjustments or performance of procedures other than those specified herein might result in hazardous radiation exposure. (C026)**

CAUTION: Data processing environments can contain equipment transmitting on system links with laser modules that operate at greater than Class 1 power levels. For this reason, never look into the end of an optical fiber cable or open receptacle. (C027)

© Copyright IBM Corp. 2022, 2024

About this publication

This publication defines, for reference purposes, the external interface of the Hardware Management Console (HMC) Web Services Application Programming Interface (Web Services API) for IBM zSystems and IBM LinuxONE, HMC version 2.16.0. This document specifies the capabilities, input and output formats, and behaviors of the Web Services API as viewed by an application external to the HMC that is leveraging that interface.

Related publications

The following publications provide information which supplements the information found within this document:

- *Capacity On Demand User's Guide*, SC28-7025
- *Processor Resource/Systems Manager Planning Guide*, SB10-7178
- *Dynamic Partition Manager (DPM) Guide*, SB10-7182
- *Small Computer Systems Interface (SCSI) IPL - Machine Loader Messages*, SC28-7029
- *3931 Installation Manual for Physical Planning*, GC28-7016
- *8561 Installation Manual for Physical Planning*, GC28-7002
- *8562 Installation Manual for Physical Planning* GC28-7011
- *3907 Installation Manual for Physical Planning*, GC28-6974
- *3906 Installation Manual for Physical Planning*, GC28-6965
- *z13 Installation Manual for Physical Planning*, GC28-6938
- *z/VM CP Planning and Administration Guide*, SC24-6178
- *z/VM CP Commands and Utility Reference*, SC24-6175
- *z/OS MVS Callable Services for HLL*, SA23-1377

Related HMC and SE console information

Hardware Management Console (HMC) and Support Element (SE) information can be found on the console help system.

Revisions

A technical change from the previous edition of this document is indicated by a thick vertical line to the left of the change.

Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

© Copyright IBM Corp. 2022, 2024

Consult assistive technologies

Assistive technology products, such as screen readers, function with the user interfaces found in this product. Consult the product information for the specific assistive technology product that is used to access our product information.

Keyboard navigation

This product uses standard Microsoft Windows navigation keys.

IBM and accessibility

See http://www.ibm.com/able for more information about the commitment that IBM® has to accessibility.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

For additional information use the following link that corresponds to your configuration:

Configuration	Link
IBM z16™ Model A02	How to send feedback to IBM
IBM z16 Rack Mount Bundle	How to send feedback to IBM
IBM LinuxONE Rockhopper 4 Model LA2	How to send feedback to IBM
IBM LinuxONE Rockhopper 4 Rack Mount Bundle	How to send feedback to IBM

Part 1. Web Services API fundamentals

Topics in this part describe the fundamentals of Web Services API.

Topics covered in this part are:

- Chapter 1, “Introduction,” on page 3
- Chapter 2, “Base definitions,” on page 55
- Chapter 3, “Invoking API operations,” on page 59
- Chapter 4, “Asynchronous notification,” on page 77
- Chapter 5, “Data model definitions,” on page 97
- Chapter 6, “Features,” on page 103

© Copyright IBM Corp. 2022, 2024

Chapter 1. Introduction

This chapter provides an overview of IBM z Unified Resource Manager (zManager) APIs, how to enable and access them, and considerations for compatibility.

Overview

The IBM z Unified Resource Manager (zManager) is a collection of advanced hardware and virtualization management functions delivered as IBM zSystems firmware. The functions of zManager are implemented as a cooperating set of components hosted on the Hardware Management Console (HMC), the Support Element (SE), and the IBM zSystems CPC. It provides a uniform, integrated administrative model for the heterogeneous computing configuration provided by an IBM zSystems environment. The functions provided by zManager include:

- Hardware inventory, initialization, configuration, monitoring and problem analysis for the components of an IBM zSystems CPC.
- Firmware installation and update for the HMC, SE, and traditional CPC components.
- Operational control and energy management for these hardware elements.

The HMC serves as the administrative access point for zManager. In that capacity, the HMC provides a web-based, remote-able graphical user interface (UI) to make the zManager functions available to users. In addition, it hosts the implementation of zManager Web Service API (Web Services API) that is described in this document.

The Web Services API is a web-oriented programming interface that makes the underlying zManager capabilities available for use by higher level management applications, system automation functions, or custom scripting. The functions that are exposed through the API support several important usage scenarios in virtualization management, including resource inventory, provisioning, monitoring, automation and workload-based optimization among others.

Components of the API

The Web Services API consists of two major components. Both components are accessed by client applications by establishing TCP/IP network connections with the HMC.

Web services interface

The web services interface is a request-and-response oriented programming interface by which client applications obtain information about the system resources managed by zManager, and by which those applications can perform provisioning, configuration or control actions on those resources.

As is the case for any web-oriented interface, client applications interact with this interface by means of the Hypertext Transfer Protocol (HTTP), an application protocol that flows over TCP/IP socket connections. Client applications request operations by forming and sending text-oriented request messages as defined by HTTP, and the Web Services API responds with text-oriented HTTP response messages. The use of HTTP makes the API client-programming-language neutral, and thus accessible to a wide variety of client applications. Client applications can be developed in programming languages such as Java, or in scripting languages such as Perl or Python that include extensive support for performing HTTP operations.

The design of the API's mapping to HTTP has been influenced by the Representational State Transfer (REST) style of interface design. The manageable resources of the system are associated with and identified by durable URIs, and the basic get, update, create and delete operations on those manageable resources are mapped directly to the HTTP GET, PUT, POST and DELETE methods. Request and response data is provided using JavaScript Object Notation (JSON), a simple, open and portable transfer representation. Mapping the functions of the API to HTTP in this way simplifies client application

© Copyright IBM Corp. 2022, 2024

development and allows access to the API without the need for extensive client side tooling or libraries as is often the case in other approaches to web services interface design.

Broadly speaking, the web services interface provides two categories of operations:

- Resource (or object) oriented operations, in which a particular request is targeted at a single manageable resource instance and typically affects just that single resource instance. The majority of the API has this orientation, for example providing functions for interacting with the virtual servers, virtualization hosts, virtual networks and workloads of the system.
- Service oriented operations, in which a particular request operates across many or all manageable resources of the system. The service-oriented operations are provided to support usage scenarios that cannot be accomplished efficiently using an object-by-object sequence of individual requests. The operations provided by the Metrics and Inventory services of the API are examples of service-oriented operations.

Asynchronous notification facility

The web services interface described above is useful to satisfy many usage scenarios, particularly those in which the client application's interest in and interaction with zManager is focused on performing a short-term task. In these kinds of applications (typical of automation or simple provisioning), the client application forms a request, gets a response, processes the response and then "forgets" about the zManager resource it interacted with. That is, the application does not attempt to retain (or cache) information about zManager resources long term and then keep that cache up to date.

However, more sophisticated management applications, including those for discovery, monitoring and advanced provisioning, are not single-request-and-forget with respect to their interest in zManager. Rather, such applications have a need to obtain and retain (i.e., cache) information about the inventory, configuration and status of many zManager resources, and to keep that cached information up to date.

In order to support these more sophisticated applications, the Web Services API provides an asynchronous notification facility by which zManager can inform interested client applications about changes to the resources managed by zManager.

The API's asynchronous notification facility provides two services to listen for messages. One is designed around the Java Message Service (JMS), an open, standard framework and API for sending messages between two or more applications. The other uses the Server-Sent Events (SSE) standard for a client to subscribe to events, which may then be streamed back to the client asynchronously over the same connection.

Base Control Program internal interface (BCPii)

Similar to the web services interface, the BCPii interface is also a request-and-response oriented programming interface by which z/OS® client applications obtain information about the system resources managed by zManager, and by which those applications can perform provisioning, configuration or control actions on those resources. This interface has also been influenced by the Representational State Transfer (REST) style of interface design, which uses the same URIs and methods as the web services interface.

Unlike the web services interface, the BCPii interface does not flow standard HTTP protocol over TCP/IP connections, instead it uses internal communications paths between the z/OS operating system and the hosting hardware. This means that the entry point for the BCPii interface is a specific hardware system, whereas the entry point for the web services interface is a Hardware Management Console. The scope of management for the web services interface is the target Hardware Management Console and the systems and resources being managed by it, while the scope for the BCPii interface is the hosting system and its resources and any other system and associated resources being managed by the Hardware Management Consoles that are managing the hosting system.

Not all services or operations are supported using the BCPii interface. Each service or operation that is supported using the BCPii interface will be documented in the specific section for that service or operation. See Appendix A, “Base Control Program internal interface (BCPii),” on page 1411 for a consolidated list of the currently supported services and operations.

Enabling and accessing the API

The Web Services API is provided on an HMC that is running with firmware version 2.11.1 or later. The API can be used to query, configure and control Central Processing Complexes (CPCs) containing a Support Element (SE) that is running with firmware version 2.11.1 or higher.

By default, the Web Services API is disabled on the HMC. When disabled, the HMC internal firewall is configured to prohibit connections to any of the TCP/IP ports used by the API. When in this state, requests to connect to the API network ports are completely ignored by the HMC without a connection-refused response.

The Web Services API can be enabled and the scope of access to it configured using the **Customize API Settings** task in the HMC UI. The **Customize API Settings** task allows an installation to enable the API through an overall enabled/disabled setting. When enabled, the HMC internal firewall is reconfigured to allow access to the relevant network ports. When the API is enabled with default settings, the HMC allows connections to the API functions from client applications accessing the HMC from any TCP/IP address. For additional security, an installation can configure the HMC to permit connections to the API ports only from selected network addresses or subnets. These addresses or subnets are specified by the **Customize API Settings** task as well. If specified, these connection restrictions are enforced by the HMC internal firewall.

In addition to the overall enablement on/off control and the optional client network address filtering, access to the API is further secured by the requirement for per-user authorization.

The HMC **User Management** task defines access and other characteristics of an HMC user. This task manages a user property (**Allow access to Web Services management interfaces**) to indicate whether a particular HMC user is to be permitted to use the API or not. By default, this setting is disabled for an HMC user profile and thus attempts to establish an API session by that user are rejected. The installation can use the **Customize API Settings** or **User Management** tasks of the HMC to set this property for one or more HMC users and thus allow those users to access the API.

Once a user is permitted to establish API sessions, its actions within those sessions are subject to the HMC's access control model, as is described in the section that follows.

For BCPii the same setup steps used for the existing, non-REST-like, interface need to be performed for the new BCPii REST-like interface. This includes enablement, community name setup and security access controls. See Appendix A, “Base Control Program internal interface (BCPii),” on page 1411 for specific details. For information about BCPii setup and installation, see https://www.ibm.com/docs/en/zos/2.4.0?topic=bcpii-setup-installation.

Authentication and access control

The HMC provides a built-in access control model in which an HMC user authenticates itself to the HMC to establish its identity, and then based on that identity is permitted or denied the ability to perform certain operations as specified by the access control configuration. These operations, and the objects on which they are permitted, are managed with object and task/action permissions that are grouped into roles that are assigned to HMC users. Roles are managed and assigned to users with the **User Management** task on the HMC.

Use of the Web Services API is subject to the same access control policy as is used for UI operations.

Establishing an API session with the HMC requires the initiating application to provide a valid HMC logon ID and corresponding password in order to authenticate and establish the identity under which its requests will be performed. (See “Logon” on page 115 for more information.) The API requires the use of SSL connections so that these login credentials can be flowed securely. The user credentials are validated by the HMC in the same way they are validated for a logon to the UI, either through the HMC's built-in user registry or by use of an LDAP directory server. If the HMC logon ID is configured to require multi-factor authentication, then an additional authentication token is required.

Once a client application has established an API session, its ability to access various managed object instances and the operations that can be performed on those instances is regulated based on the identity associated with the API session and the access control policy configured in the HMC for

those managed object instances. Access control requirements vary based on the class of managed object and the operation for the managed object. These access control requirements for API actions mirror the requirements for corresponding tasks in the HMC user interface. Details on the authorization requirements for an operation are specified in the description of that operation.

Unlike the web services interface, the BCPii interface does not use the Support Element (SE) built-in access model for authentication or access control. Instead, RACF® profiles are checked on the z/OS side to determine if the application is allowed to perform an operation against the resource specified in the request. This is similar to what is done for the existing non-REST-like BCPii interface. Additionally, the BCPii security controls can be used to control which partitions can send BCPii requests as well as which resources a partition can target with a request. Additional details can be found in Appendix A, "Base Control Program internal interface (BCPii)," on page 1411.

Compatibility

The capabilities of the Web Services API will evolve as additional management functionality is added to zManager. Over time, this evolution could result in a mixture of HMC and client application versions coexisting in a customer environment. The principles and guidelines outlined in this section are intended to maximize the compatibility and interoperability among HMC and client applications in such a mixed environment.

API versioning

Since the functionality of the Web Services API may evolve over time, each functional level of the API is identified by a version number and, beginning with version 4.10, a set of available API features. This version number is represented in major.minor form, with the initial version of the API designated as version 1.1.

The API version offered by an HMC can be determined before API logon by using the Query API Version operation (GET /api/version). The version number of the API is also provided in the response from the Logon operation. Beginning with version 4.10, the names of the available API features can be retrieved by using the `List Console API Features` and `List CPC API Features` operations. See "API features" on page 103 for more information

Enhancements to the API specification that maintain compatibility with previous versions (see principles below) are indicated by incrementing the minor portion of the version number and/or adding to or removing from the set of available API features. So, for example, the first set of compatible changes to the API would be designated as version 1.2, following the initial 1.1 version. Beginning with version 4.10, the set of available API feature would be updated.

Because the minor versions within a major version stream (e.g. the 1.x versions) are considered compatible, the HMC always offers and behaves according to the latest minor version of the API specification it supports. That means, for example, the API does not offer any facility by which a client can request version 1.1 behaviors on an HMC that offers version 1.2 level of functionality. Nor can a client request behavior tied to the presence or absence of specific API features.

While reasonable effort will be made to preserve compatibility, it may become necessary to make changes to zManager (and thus the API) that do not maintain compatibility with the previous version. If this occurs, the introduction of this new (incompatible) behavior is indicated by incrementing the major part of the version number, and starting the minor part of the version number again at 1. The first such version would thus be identified as version 2.1. It is possible for an API feature that was available to later be removed, thus it is a good practice for a client to check feature availability if it depends on the functionality provided by a certain API feature.

Allowable changes within a major version

The following kinds of changes to the API specification are allowable within a major version, and thus result in changes to the minor part of the API version number and/or the set of available API features but not to the major part of the API version number.

- Adding new object classes or new operations on existing object classes.

- Adding new properties to the data model of an existing object class.
- Changing existing properties of an existing object class from read-only or mutable to writable using the API.
- Adding new URIs and operations related to those URIs.
- Adding new optional query parameters to existing URIs where the behavior in the absence of this query parameter is unchanged.
- Adding new optional fields into input bodies where the behavior in the absence of these new fields is unchanged.
- Adding new fields to the response bodies of existing operations.
- Adding additional header or body fields to existing notification messages.
- Adding data for new classes of objects to the results provided by the Inventory service.
- Generating new types of notification messages.
- Generating property change notifications for new properties, or for existing properties that did not provide those notifications previously.
- Adding new enumeration values to enumeration-type fields returned in response bodies without removing or changing the meaning of any existing enumeration values.
- Adding new error status and reason codes.
- Adding new metric groups.
- Adding new metric fields to the end of existing metric groups.

Requirements on client applications

For a client application to correctly inter-operate with an HMC that may be offering a higher minor version of the API or additional API features, client applications must be designed and developed following the simple principle of "ignore what you don't understand" when interpreting responses or messages received from the HMC. This is necessary because the principles of allowable changes specified in the preceding section allow new fields to be added to preexisting responses or messages.

More specifically, a client application must:

- Ignore, without error, any field in a response body that is not recognized by the application.
- Ignore, without error, any header or body field in a notification message that is not recognized by the application.
- Ignore, without error, any notification message of an unrecognized type that may be received by the application.
- Ignore, without error, any object appearing in the response to an Inventory request that are of a class not recognized by the application.
- Tolerate receiving a value in a field with an enumeration data type that is an unexpected value. If the application is attempting to display this field, it might consider mapping the unrecognized enumeration value to some value indicating "other" or "unknown".
- Ignore, without error, extra values provided in a row of metric group data that reside beyond the last field currently expected by the application.

These conditions can arise as a result of API extensions that are considered allowable within a given major version. Following the "ignore what you don't understand" principle prepares a client application to tolerate these API additions should they occur.

Summary of API version updates

The following functions were introduced in the respective API version:

Note: For each of the following API version summary tables, when an API extension indicates the addition of new properties to the data model for a specified object class, such an extension also includes standard

changes to several related operations as well even though, for brevity, these related changes are not specifically mentioned in the table. In general, an extension to an object's data model will also include corresponding changes to the inputs to or responses from `Get Properties`, `Update Properties` and `Create` operations for that object class, as appropriate. In addition, the new properties are included in Inventory Service data for objects of the specified class.

Table 1. Summary of updates for API version 1.1 (HMC/SE Version 2.11.1)

Description	HMC MCL	SE MCL
Added reason codes 0, 105 and 108 as possible HTTP status code 409 (Conflict) error conditions reported by the `Migrate Virtual Server` operation.	N48180.278	N48168.275
Changed the **backing-virtualization-host-storage-resource** property of the Virtual Server data model (in the fullpack-virtual-disk nested object) from a read-only property to a writable property.	N48180.287	None
Increased the maximum request body size for the `Import Storage Access List` operation to 1 MB, and increased the maximum request body size for most other operations to 64KB.	N48180.288	N48168.294
Added the **cores-per-processor** property to the Blade object data model (read-only).	N48180.296	None
Added inventory and property-change notification support for the Virtualization Host Storage Resource object.	N48180.308	N48168.315
Added the **inventory-error-details** field and related inventory-error-info nested object to the inventory-error document returned by the `Get Inventory` operation when error condition 5 is encountered.	N48180.314	N48168.321
• Added the **properties=common** query parameter to the `Get Virtual Server Properties` operation. • Added the virtual-server-common category and power-vm-virtual-server-common, prsm-virtual-server-common, x-hyp-virtual-server-common and zvm-virtual-server-common classes to the `Get Inventory` operation.	N48180.319	N48168.325
Changed the **resources** field in the request body for the `Get Inventory` operation from required to optional.	N48180.340	None
Corrected the range checking for the **load-address** field of the `Load Logical Partition` operation so that the operation correctly supports loading from an alternate subchannel.	N48180.360	None

Table 2. Summary of updates for API version 1.2 (HMC/SE Version 2.11.1)

Description	HMC MCL	SE MCL
• Added the `Mount Virtual Media Image` operation. • Increased API version number from 1.1 to 1.2.	N48180.361	None
Corrected the format of the URI returned by the `List Members of Virtual Network` operation for a zBX TOR port to reflect the correct canonical URI format for zBX TOR port elements.	N48180.363	None
Added the **power-vm-partition-id** property to the Virtual Server object data model for PowerVM® virtual servers (read-only)	N48180.363	N48168.378
Added HTTP status code 409 (Conflict) as a possible error condition for the List Virtualization Host Storage Resources and Get Virtualization Host Storage Resource Properties operations.	N48180.376	None
Added the **feature-list** property to the Virtualization Host object data model. This property is provided for virtualization hosts on all CPCs supported by the Web Services API, but the particular features provided by a given virtualization host will differ based on the release and MCL level of the CPC.	N48180.380	N48168.402

Table 3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.0, and apply to all CPCs supported by the Web Services API: • Increased API version number from 1.2 to 1.3. • Added the **power-saving-state** property to the BladeCenter and Blade objects data models, and added the **cpc-power-saving-state** and **zcpc-power-saving-state** properties to the CPC object data model. • Added **"not-supported"** as a possible enumeration value for the power-save-allowed and power-cap-allowed properties of BladeCenter and Blade objects, and added **"not-supported"** as a possible enumeration value of the **cpc-power-save-allowed**, **cpc-power-cap-allowed**, **zcpc-power-save-allowed** and **zcpc-power-cap-allowed** properties of the CPC object. • Added the **status** (read-only), **acceptable-status** (writable), **perf-status** (read-only) and **compliant-perf-status** (writable) properties to the Workload Resource Group object data model. • Added **most-severe-perf-status** and **perf-status-data-points** fields, and related **perf-status-data-point** nested object to the response from the `Generate Workload Resource Groups Report` operation. • Added the **perf-policies** property to the Virtual Server object data model, and also added related virtual server performance policy nested object. • Added **"data-retrieval-error"** as a possible enumeration value for the status-detailed field in the response for the `Generate Load Balancing Report` operation. • Added an optional request body containing an optional **force** input field to the `Unmount Virtual Media` operation. • Changed the `Create Virtual Server` operation for a zVM virtual server to require the **password** field on input rather than allowing it to be optional. This change has been made to improve security. • Added HTTP status code 409 (Conflict) as a possible error response reported by the following Storage Management operations: – `Import Storage Access List` – `Create Virtualization Host Storage Resource` – `Delete Virtualization Host Storage Resource` – `Add Virtualization Host Storage Resource Paths` – `Remove Virtualization Host Storage Resource Paths` – `Discover Virtualization Host Storage Resources.`	H09182.023	H09173.028

Table 3. Summary of updates for API version 1.3 (HMC/SE Version 2.12.0) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.0, but apply only to CPCs with SE version 2.12.0: • Added **cp-cpu-consumption-percent**, **ifl-cpu-consumption-percent** and **other-cpu-consumption-percent** fields to the response from the `Generate Hypervisor Report` operation for zVM virtualization hosts. These new fields are provided for zVM virtualization hosts running at version 6.2 or greater. • Added the following in support of IBM zAware partitions. These changes apply only for partitions of the new **"zaware"** type: – Added **"zaware"** as a possible value of the **activation-mode** property of Logical Partition and Image Activation Profile objects. – Added the zaware-network, network-ip-info and ip-info nested objects as common nested object definitions used for new properties of Logical Partition objects. – Added the **zaware-host-name**, **zaware-master-userid**, **zaware-master-pw**, **zaware-network-info**, **zaware-gateway-info** and **zaware-dns-info** properties to the Logical Partition and Image Activation Profile object data models. – Added HTTP status 400 reason code 306 as a possible error response from the Load Logical Partition, PSW Restart, Start Logical Partition, Stop Logical Partition, and Update Image Activation Profile Properties operations when these operations are attempted on an IBM zAware partition. • Added the Cryptos and Flash Memory Adapters metric groups for CPC objects. Data entries are provided for a CPC in these metric groups if the CPC has one or more Cryptos or Flash Memory Adapters installed. • Added new **cp-cpu-time**, **ifl-cpu-time**, **zaap-cpu-time**, **ziip-cpu-time** and **icf-cpu-time** metrics to the Virtualization Host CPU and Memory metric group (for zVM virtualization hosts).	H09182.023	H09173.028
Added HTTP status code 409 (Conflict) as a possible error condition for the List Virtualization Host Storage Resources and Get Virtualization Host Storage Resource Properties operations.	H09182.062	None
Added property change support for the **unique-device-id** property of the Storage Resource object.	H09182.102	None
Added the **feature-list** property to the Virtualization Host object data model. This property is provided for virtualization hosts on all CPCs supported by the Web Services API, but the particular features provided by a given virtualization host will differ based on the release and MCL level of the CPC.	H09182.119	H09173.149

Table 4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.1 or later, and apply to all CPCs supported by the Web Services API: • Increased API version number from 1.3 to 1.4. • Added the `Get Network Adapter Properties` operation for Virtual Server objects. • Added the `List Virtualization Host Storage Resources of a Storage Resource` operation for Storage Resource objects. • Added the `List Virtual Disks of a Virtualization Host Storage Resource` operation for Virtualization Host objects. • Added API support for absolute capping to the Logical Partition and Image Activation Profile objects, including the following API extensions: – Added the **absolute-processing-capping**, **absolute-aap-capping**, **absolute-ifl-capping**, **absolute-ziip-capping** and **absolute-cf-capping** properties to the data model for Logical Partition objects. – Added the **absolute-general-purpose-capping**, **absolute-aap-capping**, **absolute-ifl-capping**, **absolute-ziip-capping** and **absolute-icf-capping** properties to the data model for Image Activation Profile elements of CPC objects. • Added the **partition-identifier** property to the data model for Logical Partition objects. The following extensions are provided by the HMC Web Services API for HMCs at version 2.12.1 or later, but primarily apply only to CPCs with SE version 2.12.1 or later:	H49574.020	H49564.021

Table 4. Summary of updates for API version 1.4 (HMC/SE Version 2.12.1) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added support for management of processor performance for virtual servers on x-hyp virtualization hosts, comprising the following API extensions: – Added the **cpu-perf-mgmt-enabled-x-hyp** property to the data model for Ensemble objects. – Added the **cpu-shares-supported** property to the data model for Virtualization Host objects. – Extended the existing **cpu-perf-mgmt-enabled**, **initial-shares**, **minimum-shares** and **maximum-shares** properties of Virtual Server objects to now also be applicable to virtual servers of type x-hyp. – Added the **workload-processor-mgmt-status**, **workload-processor-mgmt-status-reason**, **initial-shares**, **shares**, **min-shares**, and **max-shares** fields to the response for the `Generate Hypervisor Report` operation of Ensemble objects. when issued for x-hyp virtualization hosts. These fields were previously provided for PowerVM and/or z/VM® virtualization hosts but not for x-hyp virtualization hosts • Added support for ensemble availability management, comprising the following API extensions: – Added the **workload-element-groups**, **active-avail-policy**, **default-avail-policy**, **custom-avail-policies**, **avail-status**, and **compliant-avail-status** properties and related nested objects to the data model for Workload Resource Group objects. – Added the Availability Policy element of a Workload Resource Group and corresponding operations for elements of this class, including `List`, `Create`, `Delete`, `Get Properties`, `Update Properties` and `Activate` operations for Availability Policy elements of Workload Resource Group objects. – Added the Workload Element Group object and corresponding operations on objects of this class, including `List`, `Create`, `Delete`, `Get Properties` and `Update Properties` operations for Workload Element Group objects. – Added `Add To` and `Remove From` operations for managing the inclusion of Workload Element Groups within Workload Resource Group objects. – Added `List`, `Add To` and `Remove From` operations for managing the inclusion of Virtual Servers within Workload Element Group objects. – Added reporting operations for availability management, including the `Generate Workload Resource Groups Report (Ensemble Availability Management)`, `Generate Workload Resource Group Availability Status Report`, `Generate Virtual Server Report (Ensemble Availability Management)`, and `Generate Availability Status Report` operations. – Added an enumeration value of **"workload-element-group"** as a possible value of the **inclusion-type** field in the response for the `List Virtual Servers of a Workload Resource Group` operation, to specify the additional way in which virtual servers can become members of a Workload Resource Group. – Added the **avail-status**, **acceptable-avail-status**, **avail-policies** and **workload-element-group** properties and related nested objects to the data model for Virtual Server objects. • Added the **heat-load**, **heat-load-forced-air** and **heat-load-water** metrics to the zCPC Environmentals and Power metric group. • Added RoCE Adapters and Ensemble Power Metric groups.	H49574.020	H49564.021

Table 5. Summary of updates for API version 1.5 (HMC/SE Version 2.12.1)		
Description	**HMC MCL**	**SE MCL**
Added the ability to specify per-virtual-server shutdown timeouts and to perform deactivate actions that either use or override the shutdown timeout specified in the virtual server or its hosting virtualization host configuration. This new capability applies to CPCs with SE version 2.12.1 or later that have the specified MCL installed. This new capability comprises the following detailed API extensions: • Increased API version number from 1.4 to 1.5. • Added the enumeration value **"virtual-server-shutdown-timeout-override-support"** as a possible feature identifier included in the **feature-list** property of the Virtualization Host object to indicate the availability of this new capability to virtual servers hosted on a Virtualization Host instance. • Added the **shutdown-timeout-source** and **shutdown-timeout** properties to the data model of the Virtual Server object to allow a customized default shutdown timeout to be configured for a particular virtual server. • Added the **shutdown-timeout** field as an optional request body field for the `Deactivate Virtual Server` operation to allow an individual deactivation action to be performed using a shutdown timeout that is different than the timeout configured as a default for the virtual server or its hosting virtualization host.	H49574.075	H49564.070

Table 6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 1.5 to 1.6. • Added API support for managing HMC user and role definitions, comprising the following API extensions: – Added the User object representing a defined HMC user, and associated List, Create, Delete, Get Properties, Update Properties, Add User Role, and Remove User Role operations for User objects. – Added the User Role object representing a security role for HMC users, and associated List, Create, Delete, Get Properties, Update Properties, Add Permission and Remove Permission operations for User Role objects. – Added the Task object representing the permission to invoke an HMC UI task or request an associated API operation and associated List and Get Properties operations for Task objects. – Added the User Pattern object representing a pattern string used to match user IDs during logon and associated List, Create, Delete, Get Properties and Update Properties operations for User Pattern objects. – Added the `Reorder User Patterns` operation for Console objects. – Added the Password Rule object representing a password format and expiration policy specification and associated List, Create, Delete, Get Properties and Update Properties operations for Password Rule objects. – Added the LDAP Server Definition object representing the configuration of an LDAP server used for HMC authentication and associated List, Create, Delete, Get Properties and Update Properties operations for LDAP Server Definition objects. – Added the **replication-override-possible** properties for Group objects. – Added the enumeration values **"user"** and **"user-role"** as possible object class values for the Inventory Service. • Added API support for determining the characteristics of the virtual server network adapter used for GPMP purposes, comprising the following API extensions: – Added the **gpmp-network-adapter** property to the data model for Virtual Server objects. – Added the **adapter-type** property to the network-adapter-power and network-adapter-x-hyp nested object for Virtual Server objects. • Added the **gpmp-available-version** property to the data model for Virtualization Host objects. • Added the **management-enablement-level** property to the data model for CPC objects. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.0 but primarily apply only to managed system with SE version 2.13.0 or later:	N98841	CPC: N98775 zBX: N98822

Table 6. Summary of updates for API version 1.6 (HMC/SE Version 2.13.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added API support for zBX Model 004 ensemble nodes, comprising the following API extensions: – Added the **type** property to the data model for a zBX object to indicate whether the zBX is a CPC-attached (zBX Model 002/003) or ensemble node (zBX Model 004) zBX. As related extensions, also added the **type** property to the response for the `List zBXs of Ensemble` operation and also added it as an optional filtering query parameter for that operation. Note that when a zBX object represents a zBX node, the value of its **parent** property contains the URI of the Ensemble of which it is a member rather than the URI of the CPC to which it is attached. – Added an optional new input format for the `Add Node to Ensemble` operation to allow a zBX Model 004 to be specified as the node to be added to the ensemble. – Added the enumeration value **"zbx"** as a possible value for the **type** property of a Node element of an Ensemble object. – Added the **max-nodes**, **max-cpc-nodes** and **max-zbx-nodes** properties to the data model for an Ensemble object. – Added many additional properties to the data model for a zBX object when that object represents a zBX node. – Added the Get EC/MCL Description, Activate, Deactivate, Set Power Save, Set Power Capping, List Virtualization Hosts and List Virtual Servers operations for zBX node objects. – Added the `Activate` and `Deactivate` operations for BladeCenter objects contained within a zBX node. – Added the `List Virtualization Hosts` and `List Virtual Servers` operations for Ensemble nodes in general. – Added the enumeration value **"node"** as a possible value of the availability status nested object of a Virtual Server object. – Added the zBX (Node) Overview metric group as a metric group that can be requested using the Metric Service. • Added API support for manipulation of hardware messages, comprising the following API extensions: – Added the **hardware messages** container property and hardware message nested objects to the data model for zBX node and CPC objects and to the data model for the HMC Console object. – Added the List Hardware Messages, Get Hardware Message Properties and Delete Hardware Message operations for zBX node and CPC objects and the HMC Console object. • Added API support for the retrieval of audit and security log information, comprising the following API extensions: – Added a notification topic and Log Entry notification messages to provide for asynchronous push-type delivery of new log entries for the HMC Console object to interested API clients. – Added the `Get Notification Topics` operation for Session objects to allow API clients a more general way of retrieving the names of notification topics for an API session. – Added the `Get Audit Log` and `Get Security Log` operations for zBX node and CPC objects and the HMC Console object. • Added the **smt-usage**, **thread-0-usage** and **thread-1-usage** metrics in the zCPC Processors metric group. • Added the enumeration value **"virtio-scsi"** as a possible value for the emulation mode property of a virtual disk element of a Virtual Server object.	N98841	CPC: N98775 zBX: N98822

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Service API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: • Increased API version number from 1.6 to 1.7. • Added the following common request validation reason codes: – HTTP status 400 with reason code 18 – HTTP status 404 with reason codes 5 and 6 – HTTP status 409 with reason codes 4, 5, 6, and 9 • Added the following data model property qualifiers: (e) for effective properties, and (p) for pseudo properties. • Added the **effective-properties-apply** base managed object property for objects which contain an effective property (a property marked with the (e) qualifier in the object's data model). • Added API support for sending commands to and receiving messages from the operating system (OS) executing in Logical Partitions and Partitions, comprising the following API extensions: – Added the operating system notification topic, on which OS messages are received. – Added the operating system message notification message which contains the text from the OS. – Added information about any operating system notification topics associated with the API session to the Get Notification Topics response. – Added the `Open OS Message Channel` and `Send OS Command` operations for Logical Partition objects. – Added the `Open OS Message Channel` and `Send OS Command` operations for Partition objects. • Added minimal API support for Managed Virtual Machine objects, such that they can be discovered, added to User Roles and removed from User Roles, comprising the following API extensions: – Added the `List Managed Virtual Machines of a Logical Partition` operation for Logical Partition objects. The URIs returned can be used to add/remove Managed Virtual Machine objects to/from User Roles. – No further API support is provided. That is, there are no List, Create, Delete, Get Properties or Update Properties operations for Managed Virtual Machine objects, and they are not included in the Inventory Service.	P08462.053	P00339.068

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added minimal support for unmanaged CPC objects, such that they can be listed, added to User Roles and removed from User Roles. Unmanaged CPCs are those which have been discovered by the HMC but are not configured to be managed by the HMC. This support comprises the following API extensions: – Added the `List Unmanaged CPCs` operation for the Console object. The URIs returned can be used to add/remove unmanaged CPC objects to/from User Roles. – No further API support is provided. That is, there are no Create, Delete, Get Properties, or Update Properties operations for unmanaged CPC objects, and they are not included in the Inventory Service. – Added HTTP status 409 with reason code 329 on operations that target a CPC object but do not support an undefined CPC object as their target. • Added minimal support for unmanaged zBX node objects, such that they can be listed, added to User Roles, and removed from User Roles. Unmanaged zBX nodes are those which have been discovered by the HMC but are not configured to be managed by the HMC. This support comprises the following API extensions: – Added the `List Unmanaged zBX Nodes` operation for the Console object. The URIs returned can be used to add/remove unmanaged zBX node objects to/from User Roles. – No further API support is provided. That is, there are no Create, Delete, Get Properties, or Update Properties operations for unmanaged zBX node objects, and they are not included in the Inventory Service. – Added HTTP status 409 with reason code 244 on operations that target a zBX node object but do not support an undefined zBX node object as their target. • Added API support for absolute capping of Logical Partition processor usage, comprising the following API extensions: – Added the **effective-capacity**, **absolute-icf-capping**, **effective-absolute-icf-capping**, **absolute-ifl-capping**, **effective-absolute-ifl-capping**, **absolute-general-purpose-capping**, **effective-absolute-general-purpose-capping**, **absolute-ziip-capping**, **effective-absolute-ziip-capping** and **effective-properties-apply** properties to the data model for Group Profile objects. – Added HTTP status 409 with reason code 9 on `Update Group Profile Properties`.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added the **storage-total-installed**, **storage-hardware-system-area**, **storage-customer**, **storage-customer-central**, **storage-customer-expanded** and **storage-customer-available** properties to the data model for CPC objects. • Changed the data model qualifier from (w) to (wo) for the **zaware-master-pw** property of Logical Partition and Image Activation Profile objects. This is not a behavioral change but rather just a data model notation change to document the behavior using current documentation conventions. • Added the **storage-central-allocation** and s**torage-expanded-allocation** properties to the data model for Logical Partition objects. • Added a limit of 500 Workload Resource Groups per ensemble. An attempt to exceed that limit on a `Create Workload Resource Group` operation results in HTTP status 409 with reason code 66. • To improve security, the **group-profile-capacity** property of the Logical Partition object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for Logical Partition objects and it is now a read-only property. The way for an authorized API client to change a Logical Partition's **group-profile-capacity** is to change the **capacity** and/or **effective-capacity** property of the Group Profile with which the Logical Partition is associated. • Added the **effective-capacity** and **effective-properties-apply** properties to the data model for Group Profile objects. • Added HTTP status 409 with reason code 9 to `Update Group Profile Properties`. • New HTTP status 400 with reason code 330 on the `Update User Properties` operation for User Role objects when the request is to disable the API user's own user ID. • New HTTP status 400 with reason code 300 on the `Update Image Activation Profile Properties` operation for Image Activation Profile objects. • Added new **cp-all-processor-usage**, **ifl-all-processor-usage**, **icf-all-processor-usage**, and **iip-all-processor-usage** metrics to the CPC overview metric group. • Added new **exhaust-temperature-celsius** metric to the zCPC environmentals and power metric group.		
• Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. • Added several enumeration values for the **name** property of Task objects, due to new tasks added to this HMC version. • Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the **name** property of the corresponding Task objects are part of the programming API and are not changed. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 but primarily apply only to managed systems with SE version 2.13.1 or later: • Added API support for z Appliance Container Infrastructure (zACI) Logical Partitions, comprising the following API extensions, but only for Logical Partitions of the new **"zaci"** type: – Added **"zaci"** as a possible enumeration value for the **activation-mode** property of Logical Partition objects. – Added the **zaci-host-name**, **zaci-master-userid**, **zaci-master-pw**, **zaci-network-info**, **zaci-gateway-info** and **zaci-dns-info** properties to the data models for Logical Partition and Image Activation Profile objects. – Added **"zaci"** as a possible enumeration value for the **operating-mode** property of Image Activation Profile objects. • Added API support for energy optimization advice, comprising the following API extensions: – Added the `Get Energy Optimization Advice Summary` and `Get Energy Optimization Advice Details` operations for CPC objects. – Added the **last-energy-advice-time** property to the data model for CPC objects. • Added API support for absolute capping of logical partition processor usage through the new **absolute-icf-capping**, **effective-absolute-icf-capping**, **absolute-ifl-capping**, **effective-absolute-ifl-capping**, **absolute-general-purpose-capping**, **effective-absolute-general-purpose-capping**, **absolute-ziip-capping**, and **effective-absolute-ziip-capping** properties of the data model for Group Profile objects.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added API support for Dynamic Partition Manager (DPM), comprising the following API extensions (These extensions are only available if the Feature on Demand record for DPM has been installed on the Support Element): – Added the Partition object, representing a partition of a CPC, into which an operating system can be loaded and then started,and corresponding operations on object of this class, including List, Create, Delete, Get Properties, Update Properties, Start, Stop, Dump, PSW Restart, Mount ISO and Unmount ISO operations for Partition objects. – Added the Virtual Function element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties and Update Properties operations for Virtual Function elements of Partition objects. – Added the NIC element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties and Update Properties operations for NIC elements of Partition objects. – Added the HBA element of a Partition and corresponding operations for elements of this class, including Create, Delete, Get Properties, Update Properties and Reassign Storage Adapter Port for HBA elements of Partition objects. – Added the Adapter object representing a network or storage adapter, and corresponding operations on objects of this class, including List, Get Properties, Update Properties, Change Crypto Type, Create Hipersocket, Delete Hipersocket and Get Partitions Assigned to Adapter operations for Adapter objects. – Added the Network Port element of an Adapter and corresponding operations for elements of this class, including Get Properties and Update Properties for Network Port elements of Adapter objects. – Added the Storage Port element of an Adapter and corresponding operations for elements of this class, including Get Properties and Update Properties for Storage Port elements of Adapter objects. – Added the Virtual Switch object representing a CPC's network adapter and port, and corresponding operations on objects of this class, including List, Get Properties, Update Properties and Get Connected VNICs operations for Virtual Switch objects. – Added the Capacity Group element of a CPC and corresponding operations for elements of this class, including List, Create, Delete, Get Properties, Update Properties, Add Partition and Remove Partition for Capacity Group elements of CPC objects.		

Table 7. Summary of updates for API version 1.7 (HMC/SE Version 2.13.1) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd) – Added the DPM system overview, Partitions, Adapters metric groups. – Added the **dpm-enabled**, **is-cpacf-enabled** and **maximum-hipersockets** properties to the data model for CPC objects. – Added the ability to update the **description** property of CPC objects when the CPC is enabled for DPM. – Added **"dpm"** as a possible enumeration value for the **iml-mode** property of CPC objects. Note that some preexisting CPC properties are not applicable when the CPC is enabled for DPM; such properties are not returned on Get CPC Properties nor are they valid for the `Update CPC Properties` operation. – Added HTTP status 409 reason code 4 for the following operations for CPC objects, because these operations are not valid when the CPC is enabled for DPM: `Activate CPC`, `Deactivate CPC`, `Import Profiles`, and `Export Profiles`. – The following related objects are not provided for a CPC enabled for DPM: Logical Partition, Reset Activation Profile, Image Activation Profile, Load Activation Profile and Group Profile. – Added the `Start CPC`, `Stop CPC` and `Export WWPN List` operations for CPC objects. – Added API support for a partition auto-start list of a CPC, comprising the following API extensions: - Added the **auto-start-list** property to the data model for CPC objects. - Added the `Set Auto-Start List` operation for CPC objects. – Added the enumeration values **"partition"** and **"adapter"** as possible object class values for the Inventory Service. – Added the enumeration value **"dpm-resources"** as a possible object category value for the Inventory Service. • Added the **zaci-boot-selection** property to the data model for Image Activation Profile objects.		

Table 8. Summary of updates for API version 2.1 (HMC/SE Version 2.13.1)		
Description	**HMC MCL**	**SE MCL**
Increased API version number from 1.7 to 2.1. Note that the change in the major portion of the version number indicates that this version is not compatible with the previous version. The only incompatible changes in this version are due to renaming z Appliance Container Infrastructure to IBM Secure Service Container. Specifically: • The renaming of the **zaci-host-name**, **zaci-master-userid**, **zaci-master-pw**, **zaci-network-info**, and **zaci-gateway-info** Logical Partition object properties to **ssc-host-name**, **ssc-master-userid**, **ssc-master-pw**, **ssc-network-info**, and **ssc-gateway-info**, respectively. • The change of the **"zaci"** enum value for the Logical Partition object's **activation-mode** property to **"ssc"**. • The renaming of the **zaci-host-name**, **zaci-master-userid**, **zaci-master-pw**, **zaci-network-info**, **zaci-gateway-info** and **zaci-boot-selection** Image Activation Profile object properties to **ssc-host-name**, **ssc-master-userid**, **ssc-master-pw**, **ssc-network-info**, **ssc-gateway-info,** and **ssc-boot-selection**, respectively. • The change of the **"zaci"** enum value for the Image Activation Profile object's **operating-mode** property to **"ssc"**.	P08462.250	P00339.291
Removed the restriction that the **is-cpacf-enabled** property of the CPC object is only present when **dpm-enabled** is **true**. The **is-cpacf-enabled** property is now always present. Also corrected the qualifier column for **is-cpacf-enabled** to indicate that it does not provide property change notifications.	P08462.232	

Table 9. Summary of updates for API version 2.2 (HMC/SE Version 2.13.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.1 to 2.2. • Added the String/Hostname data type. • Provided a more granular value for the **sysplex-name** property of the Logical Partition to differentiate when the z/OS logical partition is not in a sysplex. The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 but primarily apply only to managed systems with SE version 2.13.1 or later: • Added API support for Secure Service Container (SSC) partitions of DPM-enabled CPCs, comprising the following API extensions: – Added the **type**, **ssc-host-name**, **ssc-boot-selection**, **ssc-ipv4-gateway**, **ssc-dns-servers**, **ssc-master-userid** and **ssc-master-pw** properties to the data model for Partition objects. – Added the **type** property to the response for the `List Partitions of a CPC` operation and also added it as an optional filtering query parameter for that operation. – Added the **ssc-management-nic**, **ssc-ip-address-type**, **ssc-ip-address**, **ssc-mask-prefix**, **vlan-id** and **mac-address** properties to the data model for NIC objects. – Added HTTP status 400 with reason codes 15, 18 and 20 to the `Create Partition` operation. – Added HTTP status 400 with reason codes 15 and 18 to the `Update Partition Properties` operation. – Added HTTP status 409 with reason code 120 to the job status for the `Start Partition` operation. – Added HTTP status 400 with reason code 15 and HTTP status 409 with reason code 8 to the `Create NIC` operation. – Added HTTP status 400 with reason code 15 to the `Update NIC Properties` operation.	P08462.261	P00339.304

Table 10. Summary of updates for API version 2.3 (HMC/SE Version 2.13.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.13.1 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.2 to 2.3. • Removed the write qualifier from the **crypto-number** property of the Adapter object. This property was never in fact writable. • Corrected the default behavior of the `Import DPM Configuration` operation to assign new object IDs, element IDs and WWPNs to import object so as to avoid the possibility of introducing objects with duplicate IDs and WWPNs. Added the **preserve-uris** and **preserve-wwpns** fields to the request body for the `Import DPM Configuration` operation to override that behavior when appropriate.	P08462.306	

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.2 to 2.20. • Added the following common request validation reason codes: – HTTP status 400 with reason codes 19 and 20 – HTTP status 409 with reason code 11 • Added API support for the IBM HMC Mobile app, comprising the following API extensions: – Added the **mobile-app-preferences** property to the Console object data model. – Added the `Get Mobile App Preferences` and `Set Mobile App Preferences` operations for the Console object to manage the console-wide mobile app settings. – Added the `Get CPC Notification Preferences for Device` and `Update CPC Notification Preferences for Device` operations for the Console object to manage the mobile app notification preferences for a CPC to a mobile device. – Added the **client-tag** field to the request body for the `Logon` operation. • Added API support to manage and interact with the console's multi-factor authentication support, comprising the following API extensions: – Added the **multi-factor-authentication-required** and **force-shared-secret-key-change** properties to the User object data model. – Added the **multi-factor-authentication-required** field to the request body for the `Create User` operation. – Added the **multi-factor-authentication-code** field to the request body for the `Logon` operation. – Added the **shared-secret-key** and **session-credential** fields to the response for the `Logon` operation. – New HTTP status 201 on the `Logon` operation. – New HTTP status 400 with reason codes 46, 47, 48 and 49 on the **Logon** operation. – Added the `Establish Shared Secret Key` operation. – Added the **hmc-time** field to the response for the `Query API Version` operation. – Expanded the authentication options when connecting to the API message broker; the session ID and session-specific credential are required for multi-factor authentication users. • Relaxed the definition of a pseudo property to no longer require a pseudo property to also be a container property. • Added the (c) data model qualifier to the **connected-vnic-uris** pseudo property of Virtual Switch objects to denote that it is a container property. Since psuedo properties were also container properties in previous versions of the API, this is a documentation-only change and not a behavioral change.	P42675.054	P42601.066

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added API enhancements that allow quicker responses when fetching the properties of certain object types, comprising the following API extensions: – Added a **properties** query parameter to the following operations to allow an API client to avoid the overhead involved with fetching object properties in which it has no interest: `Get CPC Properties`, `Get Logical Partition Properties`. – Added a **cached-acceptable** query parameter to the following operations to allow an API client to indicate that cached (but potentially out-of-date) property values are acceptable: `Get CPC Properties`, `Get Logical Partition Properties`, `Get Reset Activation Profile Properties`, `Get Image Activation Profile Properties`, and `Get Load Activation Profile Properties`. • Added the `Delete Partition Asynchronously` and `Update Partition Properties Asynchronously` operations for Partition objects. • Added the **fid** property to the data model for Virtual Function elements of Partition objects. • Made the **mac-address** property of NIC elements of Partition objects writable in certain cases. • Added the **vlan-type** property to the data model for NIC elements of Partition objects. • Relaxed the authorization requirements for access to Load Activation Profile objects through the `List Load Activation Profiles` and `Get Load Activation Profile Properties` operations to allow access without requiring CPC object-access permission under certain circumstances. This is consistent with existing HMC UI behavior. • Relaxed the authorization requirements for access to Image Activation Profile objects through the `List Image Activation Profiles` and `Get Image Activation Profile Properties` operations to allow access without requiring CPC object-access permission under certain circumstances. This is consistent with existing HMC UI behavior. • Added synchronous interfaces for retrieving operating system messages, comprising the following API extensions: – Added the `List OS Messages of a Partition` operation for Partition objects. – Added the `List OS Messages of a Logical Partition` operation for Logical Partition objects. • Added the **sequence-number** field to the os-message-info object in operating system message notifications. • Added support for more efficient access to certain Partition and Logical Partition information, comprising the following API extensions: – Added the `List Permitted Partitions` operation. – Added the `List Permitted Logical Partitions` operation. • Added the **has-unacceptable-status**, **dpm-enabled** and **se-version** properties to the response for the `List CPC Objects` operation. • Added the **classification-text** field to the response for the `Query API Version` operation. • Added the **last-used-iocds** property to the CPC object data model. • Added the **last-used-load-address** and **last-used-load-parameter** properties to the Logical Partition object data model. • Relaxed the requirements on the **boot-iso-image-name** property of Partition objects and documented the requirements in the Partition object data model. This affects the `Mount ISO Image` operation for Partition objects. • Corrected the documentation for the `Create Partition` operation; several fields were missing from the request body definition. • Added the following as a possible enumeration value for the **detected-card-type** property of an Adapter object: **"roce-express-2"**.	P42675.054	P42601.066

Table 11. Summary of updates for API version 2.20 (HMC/SE Version 2.14.0) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 but primarily apply only to managed systems with SE version 2.14.0 or later: • Added support for IBM Virtual Flash Memory, comprising the following API extensions: – Added the **storage-vfm-increment-size** and **storage-vfm-total** properties to the CPC object data model. – Added the **initial-vfm-storage**, **maximum-vfm-storage** and **current-vfm-storage** properties to the Logical Partition object data model. – Added the **initial-vfm-storage** and **maximum-vfm-storage** properties to the Image Activation Profile object data model. • Added the **port** property to the **ssc-network** nested object definition used by the Logical Partition and Image Activation Profile data models. • Extended the set of characters permitted in a load parameter, comprising the following API extensions: – Allow the following three additional characters (@, $, #) in the **load-parameter** field of the request body for the `Load Logical Partition`, `SCSI Load`, and `SCSI Dump` operations. – Allow the following three additional characters (@, $, #) in the **ipl-parameter** property in the Image Activation Profile and Load Activation Profile data models. • Removed support for controlling whether a Logical Partition entering a wait state causes termination of a time slice: – The **does-wait-state-end-time-slice** property of the CPC object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for CPC objects and it is now a read-only property whose value is always **false**. – The **end-timeslice-on-wait** property of the Reset Activation Profile object is no longer directly writable through the API. The (w) qualifier has been removed from that property in the data model for Reset Activation Profile objects and it is now a read-only property whose value is always **false**. – New HTTP status 400 with reason code 19 on the `Update CPC Properties` and `Update Reset Activation Profile Properties` operations. • Replaced the **"esa390"** and **"esa390-tpf"** logical partition activation modes with the new **"general"** activation mode, comprising the following API changes: – Added the enumeration value **"general"** as a possible value for the **activation-mode** property of a Logical Partition object. – Added the enumeration value **"general"** as a possible value for the **operating-mode** property of an Image Activation Profile object. • Due to the removal of support for dynamic changes to the SSC configuration, the SSC-related properties of the Logical Partition object are no longer directly writable through the API. The (w) qualifier has been removed from those properties in the data model for Logical Partition objects and they are now read-only properties. The affected properties are **ssc-host-name**, **ssc-master-userid**, **ssc-master-pw**, **ssc-network-info**, **ssc-gateway-info** and **ssc-dns-info**. • Added the **ssc-gateway-ipv6-info** property to the Image Activation Profile object data model. • Added the **last-energy-advice-time** property to the CPC object data model (applicable to SE version 2.13.1 or later).	P42675.054	P42601.066

Table 12. Summary of updates for API version 2.21 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.20 to 2.21. • Added the following as possible enumeration values for the **detected-card-type** property of the Adapter objects: **"osa-express-6s-1gb"**, **"osa-express-6s-10gb"**, **"osa-express-6s-1000base-t"**, **"crypto-express-6s"**, **"ficon-express-16s"**, and **"ficon-express-16s-plus"**. • Added more granular controls over which OS message notifications are presented to a mobile device, comprising the following API extensions: – Added the **new-os-message-filtered** field to the response for the `Get CPC Notification Preferences for Device` operation. – Added the **new-os-message-filtered** field to the request body for the `Update CPC Notification Preferences for Device` operation.	P42675.062	

Table 13. Summary of updates for API version 2.22 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.21 to 2.22. • Added API support for accessing the ASCII console of partitions of DPM-enabled CPCs through WebSockets, comprising the following API extension: – Added the `Get ASCII Console WebSocket URI` operation for Partition objects.	P42675.116	

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.22 to 2.23. • Added the following common request validation reason codes: – HTTP status 409 with reason codes 12 and 13 • Added support for object-specific firmware features, comprising the following API extensions: – Added the **available-features-list** property to the CPC object data model. – Added the **available-features-list** property to the Partition object data model. • Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. • Added enumeration values for the **name** property of User Role objects, due to new system-defined user roles added to this HMC version. • Added enumeration values for the **name** property of Task objects, due to new tasks added to this HMC version. • Removed enumeration values for the **name** property of Task objects for tasks that are not provided in this HMC version. • Changed the descriptive name of some classes of managed objects. Those descriptive names are not part of the programming API. The enumeration values for the managed object classes are part of the programming API and are not changed. • Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the **name** property of the corresponding Task objects are part of the programming API and are not changed. The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 but primarily apply only to managed systems with SE version 2.14.0 or later: • Added the `Import DPM Configuration` operation for CPC objects to aid in migrating a DPM configuration from a z13® system to a z14 system.	P42675.232	P42601.286

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added support for managing a FICON configuration for DPM-enabled CPCs and their Partitions, comprising the following API extensions (The majority of these extensions are only available if the **dpm-storage-management** feature is enabled on the CPC or Partition of interest): – Added the Storage Site object, representing a single storage site in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Update Properties for Storage Site objects. – Added the Storage Fabric object, representing a single storage fabric in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Update Properties for Storage Fabric objects. – Added the Storage Switch object, representing a single storage switch in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Move Storage Switch to Storage Site and Move Storage Switch to Storage Fabric for Storage Switch objects. – Added the Storage Subsystem object, representing a single storage subsystem in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Move Storage Subsystem to Storage Site, Add Connection Endpoint and Remove Connection Endpoint for Storage Subsystem objects. – Added the Storage Control Unit object, representing a single storage control unit in a FICON configuration, and corresponding operations on objects of this class, including List, Define, Undefine, Get Properties, Update Properties, Add Volume Range, Remove Volume Range for Storage Control Unit objects. – Added the Storage Path element of a Storage Control Unit and corresponding operations on elements of this class, including Create, Delete, Get Properties and Update Properties for Storage Path elements of Storage Control Unit objects. – Added the Storage Group object, representing a single storage group in a FICON configuration, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Modify Storage Group Properties, Add Candidate Adapter Ports to an FCP Storage Group, Remove Candidate Adapter Ports from an FCP Storage Group, Get Partitions for a Storage Group and Validate LUN Path for Storage Group objects. – Added the Storage Volume element of a Storage Group object and corresponding operations on elements of this class, including List, Get Properties, Fulfill FICON Storage Volume, and Fulfill FCP Storage Volume for Storage Volume elements of Storage Group objects. – Added the Virtual Storage Resource element of a Storage Group object and corresponding operations on elements of this class, including List, Get Properties, and Update Properties for Virtual Storage Resource elements of Storage Group objects. – Added the `Start Dump Program`, `Attach Storage Group to Partition`, and `Detach Storage Group from Partition` operations for Partition objects. – Added the enumeration value **"storage-volume"** as a possible value for the **boot-device** property in the Partition object data model. – Added the **boot-storage-volume** and **boot-load-parameters** properties to the Partition object data model.		

Table 14. Summary of updates for API version 2.23 (HMC/SE Version 2.14.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd) – New HTTP status 409 with reason codes 119, 120, 121 and 122 on the `Update Partition Properties` operation. – New HTTP status 409 with reason code 122 on the `Start Partition` operation. – New HTTP status 409 with reason code 12 on the following operations when the **dpm-storage-management** feature is enabled on the targeted CPC: `Export WWPN List`, `Dump Partition` and `Create HBA`. – Added the enumeration values **"fc"** and **"not-configured"** as possible values for the **type** field in the Adapter object data model. – Added the **connection-endpoint-uri** and **connection-endpoint-class** properties to the data model for Storage Port elements of Adapter objects. – Added the `Change Adapter Type` operation for Partition objects. – Added the enumeration values **"storage-site"**, **"storage-fabric"**, **"storage-switch"**, **"storage-subsystem"**, **"storage-control-unit"**, and **"storage-group"** as possible object class values for the Inventory Service. • Added support for Container Based Processors, comprising the following API extensions: – Added new **cpb-shared-processor-usage**, **cpb-dedicated-processor-usage**, and **cpb-all-processor-usage** metrics to the CPC overview metric group. – Added new **cpb-processor-usage** metric to the Logical partitions metric group. – Added **"cbp"** as a possible value for the **processor-type** metric in the zCPC processors metric group. – Added the **processor-count-cbp** and **processor-count-pending-cbp** properties to the CPC object data model. – Added the **initial-cbp-processing-weight**, **initial-cbp-processing-weight-capped**, **minimum-cbp-processing-weight**, **maximum-cbp-processing-weight**, **current-cbp-processing-weight**, **current-cbp-processing-weight-capped**, and **absolute-cbp-capping** properties to the Logical Partition object data model. – Added the **initial-cbp-processing-weight**, **initial-cbp-processing-weight-capped**, **minimum-cbp-processing-weight**, **maximum-cbp-processing-weight**, **absolute-cbp-capping**, **number-dedicated-cbp-processors**, **number-reserved-dedicated-cbp-processors**, **number-shared-cbp-processors**, and **number-reserved-shared-cbp-processors** properties to the Image Activation Profile object data model. – Added the **absolute-cbp-capping** and **effective-absolute-cbp-capping** properties to the Group Profile object data model. – Added the enumeration value **"cbp"** as a possible value for the **type** field in the Capacity Record object data model. – Added the enumeration value **"cbp"** as a possible value for the **processor-type** field in the request body for the `Add Temporary Capacity` and `Remove Temporary Capacity` operations.		

Table 15. Summary of updates for API version 2.24 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.23 to 2.24. • Added the **vendor** field to the response for the `Query API Version` operation. • Added the **maximum-partitions** property to the CPC object data model.	P42675.233	

Table 16. Summary of updates for API version 2.25 (HMC/SE Version 2.14.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.24 to 2.25. • Removed the write qualifier from the **crypto-number** property of the Adapter object. This property was never in fact writable. • Added HTTP status 409 with reason code 496 on the `Change Adapter Type` operation. • Corrected HTTP status 404 with reason code 2 to reason code 1 on the `Undefine Storage Control Unit` operation. • Added HTTP status 409 with reason code 497 to the `Create Storage Path` and `Update Storage Path Properties` operations. • Added enumeration value **"unknown"** as a possible value for the **status** field in the worldwide port name information nested object. • Added HTTP status 409 with reason code 493 to the `Create Storage Group`, `Delete Storage Group`, and `Modify Storage Group` operations. • Added object-access permission on the associated partition for `Update Virtual Storage Resource Properties` operations that update the **adapter-port-uri** or **device-number** properties. • Added HTTP status 403 with reason code 420 to the `Update Virtual Storage Resource Properties` operation. • Added HTTP status 409 with reason code 498 to the `Update Virtual Storage Resource Properties` operation.	P42675.283	

Table 17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: • Increased API version number to 2.35. • Added support to delete operating system messages, comprising the following API extensions: – Added the `Delete Partition OS Message` operation for Partition objects. – Added the `Delete Logical Partition OS Message` operation for Logical Partition objects. • Added the **last-used-world-wide port-name**, **last-used-logical-unit-number**, **last-used-disk-partition-id**, **last-used-operating-system-specific-load-parameters** and **last-used-boot-record-logical-block-address** properties to the Logical Partition object data model. • Added the **se-version** field to the response for the `List Permitted Partitions` and `List Permitted Logical Partitions` operations. • Added more detailed information about hardware messages, comprising the following API extensions: – Added the **details** property to the data model for Hardware Message elements of the Console object. – Added the **details** property to the data model for Hardware Message elements of the CPC object. – Added the **details** property to the data model for Hardware Message elements of the zBX (Node) object. • Added request aggregation services which allow what would otherwise be multiple API requests to be submitted as a single request, with their multiple results likewise returned in a single response, comprising the following API extension: – Added the `Submit Requests` operation. • Added the enumeration value **"storage-template"** as a possible object class value for the Inventory Service. • New HTTP status 400 with reason code 453 on the `Modify Storage Group` operation. • New HTTP status 400 with reason code 452 on the `Update Virtual Storage Resource Properties` operation. • New HTTP status 400 with reason code 453 on the `Create Storage Group` operation. • New HTTP status 409 with reason code 478 on the `Create Storage Group` operation. • Due to the removal of CIM actions from the HMC, the authorization requirements for some operations have changed. The affected operations are `Import Profiles`, `Export Profiles`, `Add Temporary Capacity` and `Remove Temporary Capacity` operations for CPC objects. • Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. • Added several enumeration values for the **name** property of Task objects, due to new tasks added to this HMC version. • Changed the descriptive name of some tasks. Those descriptive names are not part of the programming API. The enumeration values for the **name** property of the corresponding Task objects are part of the programming API and are not changed. • Added the following as possible enumeration values for the **detected-card-type** property of an Adapter object: **"osa-express-7s-25gb"**, **"roce-express-2-25gb"**, and **"fcp-express-32s"**. • Added **Change Object Options** task permission for `Update Logical Partition Properties` operations that update the **next-activation-profile-name** property.	P41499.052	P41414.060

Table 17. Summary of updates for API version 2.35 (HMC/SE Version 2.14.1) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: • Added the `Accept Mismatched FCP Storage Volumes` and `Reject Mismatched FCP Storage Volumes` operations for Storage Group objects. • Added the `Zeroize Crypto Domain` operation for Partition objects. • Added support for not clearing memory before loading a Logical Partition from a SCSI device, comprising the following API extensions: – Added the **clear-indicator** field to the request body for the `SCSI Load` operation for Logical Partition objects. – Allow the **clear-indicator** property of the Load Activation Profile to be set to **false** for SCSI loads. • Added support for storage group templates for DPM-enabled CPCs to facilitate the creation of storage groups with certain initial property values, comprising the following API extensions: – Added the Storage Template object, representing a single storage template, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties and Modify Properties for Storage Template objects. – Added the Storage Template Volume element of a Storage Template object and corresponding operations on elements of this class, including List and Get Properties for Storage Template Volume elements of Storage Template objects. – Added the **template-uri** field in the request body for the `Create Storage Group` operation. – Changed the **adapter-port-uri** property of the Virtual Storage Resource element object from a read-only property to a writable property.		

Table 18. Summary of updates for API version 2.36 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: • Increased API version number from 2.35 to 2.36. • Removed the write qualifier from the **crypto-number** property of the Adapter object. This property was never in fact writable. • Added HTTP status 409 with reason code 496 to the `Change Adapter Type` operation. • Corrected HTTP status 404 with reason code 2 to reason code 1 on the `Undefine Storage Control Unit` operation. • Added HTTP status 409 with reason code 497 to the `Create Storage Path` and `Update Storage Path` operations. • Added enumeration value **"unknown"** as a possible value for the **status** field in the worldwide port name information nested object. • Added HTTP status 409 with reason code 493 to the `Create Storage Group`, `Delete Storage Group`, and `Modify Storage Group` operations. • Added object access permission on the associated partition for `Update Virtual Storage Resource Properties` operations that update the **adapter-port-uri** or **device-number** properties. • Added HTTP status 403 with reason code 450 to the `Update Virtual Storage Resource Properties` operation. • Added HTTP status 409 with reason code 498 to the `Update Virtual Storage Resource Properties` operation. • Added HTTP status 400 with reason code 451 to the `Reject Mismatched FCP Storage Volumes` operation.	P41499.108	

Table 19. Summary of updates for API version 2.37 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: • Increased API version number from 2.36 to 2.37 • Added `Start FCP Storage Discovery`, `Get Connection Report`, and `Get Storage Group Histories` operations for Storage Group objects • Added a new field, **paths** to the Storage Volume element object's Data Model.	P41499.161	P41414.177

Table 20. Summary of updates for API version 2.38 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: • Increased API version number from 2.37 to 2.38 • Added Worldwide Port Name information nested object to the Virtual Storage Resource element object. • Added **"incomplete"** status to the list of WWPN **status** values in Table 240 on page 544. • Added object access permission on the referenced storage adapter for `Update Storage Path Properties` operations that update the **adapter-port-uri** property. • Added HTTP status 404 with reason code 442 to the `Create Storage Path` operation. • Added HTTP status 409 with reason code 441 to the `Update Storage Port Properties`, `Define Storage Switch`, `Create Storage Group`, `Add Candidate Adapter Ports to an FCP Storage Group`, `Fulfill FICON Storage Volume`, `Fulfill FCP Storage Volume`, `Update Virtual Storage Resource Properties`, and `Validate LUN Path` operations. • Added HTTP status 409 with reason code 457 to the `Add Connection Endpoint` operation. • Added HTTP status 409 with reason code 499 to the `Modify Storage Group Properties` operation.	P41499.208	

Table 21. Summary of updates for API version 2.39 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 but primarily apply only to managed systems with SE version 2.14.1 or later: • Increased API version number from 2.38 to 2.39. • Added API support for fulfilling multiple FICON storage volumes with a single request, comprising the following API extension: – Added the `Fulfill FICON Storage Volumes` operation to the Storage Group object.	P41499.215	

Table 22. Summary of updates for API version 2.40 (HMC/SE Version 2.14.1)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.14.1 and apply to all SE versions supported by the Web Services API: • Increased API version number to 2.40. • Corrected the handling of the **properties** query parameter of the `Get CPC Properties` and `Get Logical Partition Properties` operations to no longer return HTTP status 400 for properties the HMC supports but the target CPC or Logical Partition does not support. Such properties are simply omitted from the response.	P41499.257	

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.1. Note that the change in the major portion of the version number indicates that this version is not compatible with the previous version. The vast majority of the incompatibilities are due to the removal of zManager's support for IBM z BladeCenter Extension (zBX), the blades of a zBX, alternate HMC, Ensembles, and related components. More detailed information is provided below. • Removed support for IBM z BladeCenter Extension (zBX), the blades of a zBX, alternate HMC, Ensembles, and related components. The majority of that support is described in *Part 4: "Ensemble and zBX management"* of the *Hardware Management Console Web Services API*, SC27-2637. The changes are: – Removed support for the following managed object and element object classes and types. This includes removal of all operations, inventory classes, inventory categories, and metric groups related to these object classes and types. - ensemble - node - zbx - top-of-rack-switch - rack - bladecenter - blade - virtualization-host - virtual-switch - virtual-server - storage-resource - virtualization-host-storage-resource - virtualization-host-storage-group - network adapter of a virtual server - virtual disk - performance policy - availability policy – Removed all operations described in *Part 4: "Ensemble and zBX management"* of the *Hardware Management Console Web Services API*, SC27-2637. As that is a considerable number of operations, they are not listed here. – Removed the following inventory categories and all classes contained therein from the Inventory Service: - zvm-resources - power-vm-resourses - x-hyp-resources - prsm-resources - virtual-server-common - zbx-resources - ensemble-wide-resources	P46683.065	P46598.076

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd.) – Removed the following metric groups from the Metrics Service: - BladeCenter temperature and power metric group - Blade power - zBX (Node) overview - Blade CPU and memory metric group - Ensemble power - Virtual server CPU and memory metrics group - Virtualization host CPU and memory metrics group - Workload service class data metrics group - Virtualization host (vSwitch) uplink metric group - Virtualization host (vSwitch) by virtual network metric group - Attached virtual server network adapters metric group - Optimizer IEDN virtual network interface metric group - Optimizer IEDN physical network adapter metric group - Top-of-rack switch ports metrics - ESM switch port metrics – Removed the following operations not previously described in the *"Ensemble and zBX management"* section: - Make Console Primary - List Ensemble CPC Objects - List Unmanaged zBX Nodes – Removed the following object properties: - Removed the **paired-role**, **is-auto-switch-enabled**, **paired-hmc**, and **ip-swapping-available** properties of the Console object - Removed **is-ensemble-member** and **management-enablement-level** properties of the CPC object – Removed several object types from those eligible for membership in pattern-matching groups. The following are no longer valid enumeration values for the **types** property of a Group object: - **"zvm-virtual-machines"** - **"blade-center"** - **"data-power-xi50z-blades"** - **"ibm-smart-analytics-optimizer-blades"** - **"power-blade"** - **"system-x-blade"** - **"power-vm-virtual-server"** - **"system-x-virtual-server"** - **"workload"** - **"defined-zbx-node"** - **"director-timer-console"** - **"ibm-fiber-saver"**		

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd.) – Removed several object types from those that can be permitted in user roles. The following are no longer valid enumeration values for the **permitted-object** property of a User Role object: - **"alternate-console"** - **"blade-center"** - **"datapower-device"** - **"default-workload"** - **"eckd-storage-resource"** - **"ensemble"** - **"fcp-storage-resource"** - **"p-blade"** - **"p-virtual-server"** - **"undefined-zbx-node"** - **"virtual-network"** - **"workload-element-group"** - **"workload-resource-group"** - **"x-blade"** - **"x-virtual-server"** - **"zbx-node"** - **"zvm-fcp-storage-resource"** - **"zvm-virtual-machine"** – Removed several system-defined User Roles. User Roles with the following enumeration values for the **name** property have been removed: - **"hmc-bladecenter-objects"** - **"hmc-dpxi50z-blade-objects"** - **"hmc-ensemble-administrator-tasks"** - **"hmc-ensemble-object"** - **"hmc-ibm-blade-objects"** - **"hmc-ibm-blade-virtual-server-objects"** - **"hmc-policy-administrator-tasks"** - **"hmc-policy-operator-tasks"** - **"hmc-storage-resource-administrator-tasks"** - **"hmc-storage-resource-objects"** - **"hmc-virtual-network-administrator-tasks"** - **"hmc-virtual-network-objects"** - **"hmc-virtual-server-administrator-tasks"** - **"hmc-virtual-server-operator-tasks"** - **"hmc-workload-administrator-tasks"** - **"hmc-workload-objects"** - **"hmc-z-vm-virtual-machine-objects"** - **"hmc-z-vm-virtual-machine-tasks"** – Removed all tasks related to zBX and Ensemble components. Enumeration values for the **name** property of those Task objects are not provided in this HMC version. As that is a considerable number of tasks, they are not listed here.		

Table 23. Summary of updates for API version 3.1 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd.) • Added the `Resend Request` operation for Storage Group objects. • Added the enumeration value **"auto-created"** as a possible value for the **action** field of a storage-group-action-info nested object related to Storage Group histories. • Corrected HTTP status 404 with reason code 2 to be reason code 1 on the `Undefine Storage Switch` operation. • Added support for more granular action permissions for the HMC Mobile app, comprising the following API extensions: – Added several new action settings properties, one for each action that supports the new granular permission settings, to the mobile-app-preferences nested object of the Console object. These settings are updated with the `Set Mobile App Preferences` and included in the response to the `Get Mobile App Preferences` operation. – New HTTP status 400 with reason code 337 and HTTP status 404 with reason code 2 on the `Set Mobile App Preferences` operation – Added the **user-template-uri** property to the User object data model • Added the ability for an API client to request cancellation of the `Start Partition`, `Stop Partition` and `Start CPC` operations with the `Cancel Job` operation.		
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added detailed information about the processors in a logical partition, including the number of allocated and reserved processors of various types and whether they are dedicated or shared. Added the **processor-usage**, **number-general-purpose-processors**, **number-reserved-general-purpose-processors**, **number-general-purpose-cores**, **number-reserved-general-purpose-cores**, **number-icf-processors**, **number-reserved-icf-processors**, **number-icf-cores**, **number-reserved-icf-cores**, **number-ifl-processors**, **number-reserved-ifl-processors**, **number-ifl-cores**, **number-reserved-ifl-cores**, **number-ziip-processors**, **number-reserved-ziip-processors**, **number-ziip-cores**, and **number-reserved-ziip-cores** properties to the Logical Partition object data model. • Added the ability to increase the number of allocated and reserved processor cores in a logical partition through the APIs, comprising the following API extensions: – Added writable properties containing the number of allocated or reserved cores for various processor types to the Logical Partition data model. – Add the authorization requirement of action/task permission for the **Logical Processor Add** task when increasing the number of allocated or reserved cores. • Added support to control whether Elliptic Curve Cryptography (ECC) keys can be used for computing digital signatures, comprising the following API extensions: – Added the **permit-ecc-key-import-functions** property to the Image Activation Profile object data model • Added support to determine whether System Recovery Boost is enabled for a logical partition, comprising the following API extensions: – Added the is-sub-capacity-boost-active property to the Logical Partition object data model • Added support to determine whether zIIP Boost is enabled a logical partition, comprising the following API extensions: – Added the **is-ziip-capacity-boost-active** property to the Logical Partition object data model – Added the enumeration value **"boost"** as a possible value for the **record-type** property of a Capacity Record element of a CPC object. – Added the **maximum-real-hours**, **maximum-test-hours**, **remaining-real-hours** and **remaining-test-hours** properties to the data model for Capacity Record elements of CPC objects.	P46683.065	P46598.076

Table 24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 3.1 to 3.2. • Added API support to manage and interact with the console's multi-factor authentication support provided in conjunction with MFA servers, comprising the following API extensions: – Added the MFA Server Definition object representing the configuration of an MFA server used for HMC authentication and associated List, Create, Delete, Get Properties and Update Properties operations for MFA Server Definition objects. – Added the **mfa-types**, **primary-mfa-server-definition-uri**, **backup-mfa-server-definition**, **mfa-policy**, **mfa-userid** and **mfa-userid-override** properties to the User object data model. – Added the **mfa-types**, **primary-mfa-server-definition-uri**, **backup-mfa-server-definition**, **mfa-policy**, **mfa-userid** and **mfa-userid-override** fields to the request body for the Create User operation. – Added the **rsa-securid-passcode** field to the request body for the `Logon` operation. – Added the **shared-secret-key** and **session-credential** fields to the response for the `Logon` operation. – New HTTP status 400 with reason code 337 and HTTP status 404 with reason codes 338 and 339 on the `Create User` operation. – New HTTP status 400 with reason code 337, HTTP status 404 with reason codes 338 and 339, and HTTP status 409 with reason code 337 on the `Update User Properties` operation. – New HTTP status 400 with reason code 53, HTTP status 409 with reason codes 50, 51 and 52, HTTP status 500 with reason code 40, and HTTP status 503 with reason code 40 on the `Logon` operation. – New HTTP status 409 with reason code 51 on the `Establish Shared Secret Key` operation. – Added the `Provide Requested MFA Information` operation. – Added the `Change Logon Password` operation. • Added API support for fulfilling multiple FICON storage volumes with a single request, comprising the following API extension: – Added the `Fulfill FICON Storage Volumes` operation to the Storage Group object. • Added support for importing a DPM configuration from a z14 system on which the **"dpm-storage-management"** feature is enabled, comprising the following API extensions: – Added the following fields to the request body for the Import DPM Configuration operation for CPC objects: **storage-sites**, **storage-subsystems**, **storage-fabrics**, **storage-switches**, **storage-control-units**, **storage-paths**, **storage-groups**, **storage-volumes**, **storage-templates**, **storage-template-volumes**, **virtual-storage-resources**, **storage-ports**, and **network-ports**. – New HTTP status code 200 on the `Import DPM Configuration` operation. When HTTP status code 200 is returned a response body is provided that contains information about the parts of the configuration that were not restored.	46683.097	P46598.116, P46650.003

Table 24. Summary of updates for API version 3.2 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added support to control the enablement of IBM Secure Boot for Linux®, comprising the following API extension: – Added the **secure-boot** property to the Partition object data model – Added the **secure-boot** property to the Image Activation Profile object data model – Added the **secure-boot** property to the Load Activation Profile object data model – Added the **last-used-secure-boot** property to the Logical Partition object data model – Added the **secure-boot** field to the request body for the SCSI Load operation for Logical Partition objects. – Added the **secure-boot** field to the request body for the SCSI Dump operation for Logical Partition objects. • Added support to control whether Elliptic Curve Cryptography (ECC) keys can be used for computing digital signatures, comprising the following API extensions: – Added the **permit-ecc-key-import-functions** property to the Partition object data model. • New HTTP status 400 with reason code 20 on the `Update Partition Properties` operation.		

Table 25. Summary of updates for API version 3.3 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number from 3.2 to 3.3. • New HTTP status 400 with reason code 54 on the `Logon` operation. • Added a note to the **workload-manager-enabled** property of Logical Partition object to point out that changing that property may have no effect in some situations.	46683.098	
• Removed the checking for unrecognized **X-*** HTTP request headers, and, therefore, removed the following common request validation reason code: HTTP status 400 with reason code 10.	46683.099	
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • New HTTP status 409 with reason 130 and HTTP status 500 with reason 130 on the `Start Partition` operation. • Added the optional **bootloader-error-id** field to the **job-results** object in the asynchronous result of the `Start Partition` operation.	46683.102	46598.119

Table 26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.4. • Added the following as possible enumeration values for the **vendor** field in the response body for the `Query API Version` operation: **"d"**, **"e"**, **"f"**, **"g"**, **"h"** and **"i"**. • New HTTP status 403 with reason code 40 on the `Provide Requested MFA Information` operation. • Added support for cloud network adapters, comprising the following API extensions: – Added the enumeration value **"cna"** as a possible value for the **type** property of an Adapter object. – Added the enumeration value **"cna"** as a possible value for the **adapter-family** property of an Adapter object. – Added the following as possible enumeration values for the **detected-card-type** property of an Adapter object: **"ficon-express-16sa"**, and **"cloud-network-x5"**. – Added the enumeration value **"cna"** as a possible value for the **type** property of a NIC element of a Partition object. – Added the **function-number** and **function-range** properties to the data model for NIC elements of Partition objects. – Added the **function-number** and **function-range** fields to the request body for the `Create NIC` operation. – New HTTP status 409 with reason code 18 on the `Update NIC Properties` operation. • New HTTP status 409 with reason code 125 on the `Start Partition` operation. • Documented that the **template-uri** and **type** fields of the request body for the `Create Storage Group` operation are mutually exclusive. • Added support to request service for Console Hardware Messages, comprising the following API extensions: – Added the `Request Console Service`, `Get Console Service Request Information` and `Decline Console Service` operations for the Console object. – Added the **service-supported** property to the data model for the hardware-message nested object of the Console object. • Added support to request service for CPC Hardware Messages, comprising the following API extensions: – Added the `Request CPC Service`, `Get CPC Service Request Information` and `Decline CPC Service` operations for the CPC object. – Added the **service-supported** property to the data model for the hardware-message nested object of CPC objects. • Added additional information about degraded adapters, comprising the following API extensions: – The **degraded-adapters** property of Partition objects may now include Virtual Storage Resource element URIs. – Added the **degraded-reasons** property to the data model for Virtual Storage Resource elements of Storage Group objects.	P46683.160	P46598.194

Table 26. Summary of updates for API version 3.4 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added support to determine whether Secure Execution for Linux is enabled for a logical partition or partition, comprising the following API extensions: – Added the **is-secure-execution-enabled**, **is-global-key-installed** and **is-host-key-installed** properties to the CPC object data model. – Added the **is-secure-execution-enabled** property to the Logical Partition object data model. – Added the **secure-execution** property to the Partition object data model. • Added the Power Status metric group for CPC objects. • Added the **zcpc-minimum-inlet-air-temperature**, **zcpc-maximum-inlet-air-temperature**, **zcpc-maximum-inlet-liquid-temperature** and **zcpc-environmental-class** properties to the CPC object data model.		

Table 27. Summary of updates for API version 3.5 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.5. • Corrected the handling of the **properties** query parameter of the `Get CPC Properties` and `Get Logical Partition Properties` operations to no longer return HTTP status 400 for properties the HMC supports but the target CPC or Logical Partition does not support. Such properties are simply omitted from the response.	P46683.169	

Table 28. Summary of updates for API version 3.6 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.6. • Removed the requirement for object-access permission to the logical partition's parent CPC object from the following Logical Partition operations: Activate Logical Partition, Deactivate Logical Partition, Reset Normal, Reset Clear, Load Logical Partition, PSW Restart, Start Logical Partition, Stop Logical Partition, SCSI Load, and SCSI Dump. The authorization requirements for these operations now matches the requirements for the corresponding HMC UI tasks, and they allow for more granular control of permissions relating to Logical Partitions and CPCs. • Added support for the request service action permission for the HMC Mobile app, comprising the following API extensions: – Added the **action-settings-request-service-hardware-message** property to the mobile-app-preferences nested object of the Console object. • Added the ability to control whether a Hardware Message is deleted when requesting service for it and to obtain customer contact information for a Hardware Message, comprising the following API extensions: – Added the **delete** query parameter to the Get Console Service Request Information and Get CPC Service Request Information operations. – Added the **customer-name** and **customer-phone** fields to the response body for the Get Console Service Request Information and Get CPC Service Request Information operations. • Added additional information to storage group histories, comprising the following API extensions: – Added the **storage-group-name** and **storage-group-type** query parameters to the Get Storage Group Histories operation. – Added the **storage-group-name** and **storage-group-type** fields to the storage-group-history-info nested object in the response body of the Get Storage Group Histories operation.	P46683.196	

Table 29. Summary of updates for API version 3.7 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.7. • Added the **storage-volumes-in-error** and **storage-volumes-recovered** fields to the storage-group-configuration nested object in the response body of the `Get Storage Group Histories` operation. • Added the **adapter-uri**, **old-serial-number**, and **new-serial-number** fields to the storage-volume-info nested object in the response body of the `Get Storage Group Histories` operation. • Added the **last-used-load-type** property to the Logical Partition object data model. • The **last-used-load-address** property of the Logical Partition object may be now 4 or 5 characters in length. Previously, it was always 5 characters, but it is 4 characters for NVMe load addresses. • The **load-parameter** request body field of the `Load Logical Partition`, `SCSI Load`, and `SCSI Dump` operations may now be 0-length. Previously, it was required to be 1-256 characters. • The **operating-system-specific-load-parameters** request body field of the `SCSI Load` and `SCSI Dump` operations may now be 0-length. Previously, it was required to be 1-256 characters. • Added the **force** field to the request body for the `SCSI Dump` operation. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added support for Non-volatile Memory Express (NVMe) adapters, comprising the following API extensions: – Added the enumeration value **"nvme"** as a possible value for the **type** property of an Adapter object. – Added the enumeration value **"nvme"** as a possible value for the **adapter-family** property of an Adapter object. – Added the **ssd-is-installed**, **ssd-capacity**, **ssd-model-number**, **ssd-serial-number**, **ssd-subsystem-vendor-id**, and **ssd-vendor-id** properties to the Adapter object data model. – Added the enumeration value **"nvme"** as a possible value for the **type** property of a Storage Group object. – Added the **adapter-uri**, **serial-number**, and **fid** properties Storage Volume elements of Storage Group objects. – New HTTP status 400 with reason code 452, HTTP status 404 with reason code 446, HTTP status 409 with reason code 446, HTTP status 409 with reason code 500 on the `Create Storage Group` operation. – New storage-volume-request-info nested object format for **"create"** and **"modify"** operations on NVMe storage volumes, for use on the `Modify Storage Group Properties` operation. – New HTTP status 400 with reason code 452, HTTP status 409 with reason code 446, HTTP status 409 with reason code 500 on the `Modify Storage Group` operation. – New HTTP status 404 with reason code 4 on the `Resend Request` operation. – New HTTP status 404 with reason code 4 on the `Accept Mismatched Storage Volumes` operation. – Added the enumeration values **"nvme-ssd-removed"**, **"nvme-ssd-installed"**, **"nvme-volume-mismatch"**, and **"nvme-volumes-accepted"** as a possible value for the action field of the storage-group-action-info nested object in the response body of the `Get Storage Group Histories` operation. – Added the `NVMe Load` and `NVMe Dump` operations for Logical Partition objects. – Added the enumeration values **"ipltype-nvmeload"** and **"ipltype-nvmedump"** as a possible value for the **ipl-type** property of an Image Activation Profile object. – Added the enumeration values **"ipltype-nvmeload"** and **"ipltype-nvmedump**" as a possible value for the **ipl-type** property of a Load Activation Profile object.	P46683.234	P46598.279

Table 30. Summary of updates for API version 3.8 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.8. • Documented HTTP status 409 on the `Update Logical Partition Properties` operation. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added the **global-primary-key-hash**, **global-secondary-key-hash**, **host-primary-key-hash**, and **host-secondary-key-hash** properties to the CPC object data model.	P46683.243	P46598.318

Table 31. Summary of updates for API version 3.9 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.9. • Added the Remote Firmware Update element object to the Console object. • Added the `List Remote Firmware Updates of the Console`, `Get Console Remote Firmware Update Properties`, `Delete Console Remote Firmware Update`, and `Authorize Remote Firmware Updates` operations for the Console object. • Added enumeration values for tasks within User Role objects. • Added the **sna-name** property to the CPC object data model. • Added the **sna-name** property to the Console object data model. • Added the **has-important-unviewed-operating-system-messages** property to the Logical Partition object data model. • Added **group-uri** query parameter to the `Get CPC Properties` operation for the CPC object. • Added **group-uri** query parameter to the `Update CPC Properties` operation for the CPC object. • Added **group-uri** query parameter to the `Get Logical Partition Properties` operation for the Logical Partition object. • Added **group-uri** query parameter to the `Update Logical Partition Properties` operation for the Logical Partition object. • Added the **is-held**, **is-priority**, and **max-messages** query parameters to the `List OS Messages of a Logical Partition` operation for the Logical Partition object. • Added the `Get Console Events Log` operation for the Console object. • Added the **event-id** and **max-entries** query parameters to the following operations: – `Get Console Audit Log` operation for the Console object. – `Get Console Security Log` operation for the Console object. – `Get CPC Audit Log` operation for the CPC object. – `Get CPC Security Log` operation for the CPC object. • Added the `Get CPC Events Log`, `Get LPAR Resource Assignments`, `Get LPAR Controls`, and `Update LPAR Controls` operations for the CPC object. • Added **properties** query parameter to the `Get Console Properties` operation for the Console object. • Added **properties** query parameter to the `Get Reset Activation Profile Properties` operation for the Reset Activation Profile object. • Added **properties** query parameter to the `Get Image Activation Profile Properties` operation for the Image Activation Profile object. • Added **properties** query parameter to the `Get Load Activation Profile Properties` operation for the Load Activation Profile object. • Added **properties** query parameter to the `Get Group Profile Properties` operation for the Group Profile object. • Added the **partition-identifier** property to the Image Activation Profile object data model. • Added the **logical-partition-names** and **effective-logical-partition-names** properties to the Group Profile object data model. • Added **properties** query parameter to the `Get Capacity Record Properties` operation for the Capacity Record object.	P46683.309 and P46686.001, Bundle H25	P46598.370 and P46613.001, Bundle S38

Table 31. Summary of updates for API version 3.9 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added support to determine which user template to use during logon processing based on the user's LDAP group membership, comprising the following API extensions. – Added the **specific-template-uri**, **template-name-override-ldap-server-definition-uri**, **template-name-override-default-template-uri**, **ldap-group-to-template-mappings**, **ldap-group-ldap-server-definition-uri**, **ldap-group-default-template-uri**, and **domain-name-restrictions-ldap-server-definition-uri** properties to the data model for User Pattern elements of Console objects. – Added the **specific-template-uri**, **template-name-override-ldap-server-definition-uri**, **template-name-override-default-template-uri**, **ldap-group-to-template-mappings**, **ldap-group-ldap-server-definition-uri**, **ldap-group-default-template-uri**, and **domain-name-restrictions-ldap-server-definition-uri** fields to the request body for the Create User Pattern operation. – New HTTP status 404 with reason codes 343, 344, 345, 346, 347, 348, and 349 on the `Create User Pattern` operation. – New HTTP status 404 with reason codes 343, 344, 345, 346, 347, 348, and 349 on the `Update User Pattern Properties` operation. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added the Remote Firmware Update element object to the CPC object. • Added the `List Remote Firmware Updates of the CPC`, `Get CPC Remote Firmware Update Properties`, and `Delete CPC Remote Firmware Update` operations for the CPC object. • The following object classes and operations are now supported using the BCPii interface: – Session management services: - `Query API Version` – Asynchronous job processing: - `Query Job Status`, `Delete Completed Job Status`, `Cancel Job` – Console object: - `Get Console Properties`, `Restart Console`, `Shutdown Console`, `Get Console Audit Log`, `Get Console Security Log`, `Get Console Events Log`, `List Hardware Messages`, `Get Console Hardware Message Properties`, `Delete Console Hardware Message`, `Requet Console Service`, `Get Console Service Request Information`, and `Decline Console Service`. – Group object: - `List Custom Groups`, `Get Custom Group Properties`, `Create Custom Group`, `Delete Custom Group`, `Add Member to Custom Group`, `Remove Member from Custom Group`, and `List Custom Group Members`. – CPC object: - `List CPC Objects`, `Get CPC Properties`, `Update CPC Properties`, `Activate CPC`, `Deactivate CPC`, `Import Profiles`, `Export Profiles`, `Add Temporary Capacity`, `Remove Temporary Capacity`, `Swap Current Time Server`,`Set STP Configuration`, `Change STP-only Coordinated Timing Network`, `Join STP-only Coordinated Timing Network`, `Leave STP-only Coordinated Timing Network`, `List CPC Hardware Messages`, `Get CPC Hardware Message Properties`, `Delete CPC Hardware Message`, `Request CPC Service`, `Get CPC Service Request Information`, `Decline CPC Service`, `Get CPC Audit Log`, `Get CPC Security Log`, `Get CPC Events Log`, `Get LPAR Resource Assignments`, `Get LPAR Controls`, and `Update LPAR Controls`.		

Table 31. Summary of updates for API version 3.9 (HMC/SE Version 2.15.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• (cont'd.) – Logical Partition object: - `List Logical Partitions of a CPC`, `List Permitted Logical Partitions`, `Get Logical Partition Properties`, `Update Logical Partition Properties`, `Activate Logical Partitions`, `Deactivate Logical Partition`, `Reset Normal`, `Reset Clear`, `Load Logical Partition`, `PSW Restart`, `Start Logical Partition`, `Stop Logical Partition`, `Send OS Command`, `List OS Message of a Logical Partition`, `Delete Logical Partition OS Message`, `SCSI Load`, `SCSI Dump`, `NVMe Load`, and `NVMe Dump`. – Reset activation profile: - `List Reset Activation Profiles`, `Get Reset Activation Profile Properties`, and `Update Reset Activation Profile Properties`. – Image activation profile: - `List Image Activation Profiles`, `Get Image Activation Profile Properties`, and `Update Image Activation Profile Properties`. – Load activation profile: - `List Load Activation Profiles`, `Get Load Activation Profile Properties`, and `Update Load Activation Profile Properties`. – Group profile: - `List Group Profiles`, `Get Group Profile Properties`, and `Update Group Profile Properties`. – Capacity records: - `List Capacity Records` - `Get Capacity Record Properties` – Energy Management: - `Set CPC Power Save`, `Set CPC Power Capping`, `Set zCPC Power Save`, `Set zCPC Power Capping`, `Get CPC Energy Management Data`, `Get Energy Optimization Advice Summary`, and `Get Energy Optimization Advice Details`. • Added the **target-name** property to the Console object data model. • Added the **primary-se** property to the network-info object returned from the `Get Console Properties` operation of the Console object. • Added the **name** property to the detailed-network-info object returned from the `Get Console Properties` operation of the Console object. • Added the **request-origin** property to the logical-partition-info object returned from the `List Logical Partitions of a CPC` and `List Permitted Logical Partitions` operations of the Logical Partition object. • Added the **target-name** property to the CPC object data model. • Added the **target-name** property to the Logical Partition object data model. • Added the **target-name** property to the Reset activation profile object data model. • Added the **target-name** property to the Image activation profile object data model. • Added the **target-name** property to the Load activation profile object data model. • Added the **target-name** property to the Group profile object data model. • Added the **target-name** property to the Capacity records object data model. • Added the **request-origin** property to the Logical Partition object data model. • Added the enumeration value **acceptable** as a possible **status** value for the CPC object. • Added **target-name** and **location** properties to the cpc-info object returned for the `List CPC Objects` operation of the CPC object. • Added the enumeration value **acceptable** as a possible **status** value for the Logical Partition object. • Added HTTP status 403 with reason code 0 on operations using the BCPii interface from source partitions that do not have the necessary BCPii security controls permission.		

Table 32. Summary of updates for API version 3.10 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.10. • Added enumeration values for classes of managed objects within User Role objects, due to new managed object types added to this HMC version. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.14.1 or later: • Added support for managing an FCP tape library and related components for DPM-enabled CPCs and their partitions, comprising the following API extensions. (The majority of these extensions are only available if the **dpm-fcp-tape-management** feature is enabled on the CPC or Partition of interest): – Added the Tape Library object, representing a single physical tape storage unit associated with a DPM-enabled CPC, and corresponding operations on objects of this class, including List, Undefine, Get Properties, Update Properties, Request Tape Library Zoning, and Discover Tape Libraries, for Tape Library objects. – Added the Tape Link object, representing a single tape link to a tape library, and corresponding operations on objects of this class, including List, Create, Delete, Get Properties, Modify Tape Link Properties, Add Adapter Ports, Remove Adapter Ports, Replace Adapter Port, Resend Request, Get Partitions for a Tape Link, Get Tape Link Histories, Update Tape Link Environment Report, and Get Tape Link Environment Report, for Tape Link objects. – Added the Virtual Tape Resource element of a Tape Link object and corresponding operations on elements of this class, including List, Get Properties, and Update Properties for Virtual Tape Resource elements of Tape Link objects. – Added the enumeration values **"tape-link"** and **"tape-library"** as possible object class values for the Inventory Service. – Added the **tape-link-uris** property to the Partition object data model. – Added the Attach Tape Link to Partition and Detach Tape Link from Partition operations, for Partition objects. – Added the **management-world-wide-port-name** property to the CPC object data model. – Added the enumeration value **"dpm-fcp-tape-management"** as a possible value for the **name** property of the cpc-feature-info nested object in the CPC object data model. – Added the enumeration value **"dpm-fcp-tape-management"** as a possible value for the **name** property of the partition-feature-info nested object in the Partition object data model. • New HTTP status 409 with reason code 153 on the Detach Storage Group from Partition operation.	P46683.309, Bundle H25	For z15® Systems, P46598.369 and P46612.001, Bundle S38, are required. For z14 Systems, P41414.332, Bundle S59, is required.

Table 33. Summary of updates for API version 3.11 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.11. • Removed the (pc) qualifier from the Tape Link **incomplete-reasons** property. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Added the **node-name** and **target-name** fields to the stp-node shared nested object table.	P46683.311, Bundle H26	P46598.376, Bundle S40

Table 34. Summary of updates for API version 3.12 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.12. • Corrected an error that caused the remote firmware update operations to end with an internal server error. • Corrected the documentation to add reason code 2 as a possible HTTP status code 404 (Not Found) error condition reported by the `Create User Role` operation. • Clarified the descriptions for the **cp-processors** and **ifl-processors** properties in the Partition object to state that partitions can have only CPs or only IFLs but not a mix of both. The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 but primarily apply only to managed systems with SE version 2.15.0 or later: • Clarified the qualifier information for the **access-coprocessor-group-set** property of the Partition object to indicate that when the **se-version** property of the associated CPC is **"2.15.0"** or later, this property is not permitted on an `Update Partition Properties` operation, and its value is always false. • Added reason code 19 as a possible HTTP status code 400 (Bad Request) error condition reported by the `Update Partition Properties` operation.	P46683.408 (Bundle H49)	

Table 35. Summary of updates for API version 3.13 (HMC/SE Version 2.15.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.15.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 3.13. • Added the **action-settings-manage-remote-firmware-updates** field to the request body of the `Set Mobile App Preferences` operation. • Added the **action-settings-manage-remote-firmware-updates** property to the mobile-app-preferences nested object in the Console object's data model. • Added the **partition-links** property to the request body contents of the `Import DPM Configuration` operation, and added information to reason code 7 of HTTP status code 400 (Bad Request) to state that the Firmware Feature enablement of the designated CPC does not support the provided configuration objects. • Added reason code 8 to HTTP Status code 409 (Conflict) as a possible error response from the `Update Group Profile Properties` operation.	P46683.408 (Bundle H49)	

Table 36. Summary of updates for API version 4.1 (HMC/SE Version 2.16.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.16.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 4.1. Note that the change in the major portion of the version number indicates that this version is not compatible with the previous version. The majority of these incompatibilities are due to the removal of zManager's support for Container Based Processors. For security reasons, several specific HTTP status codes and reason codes have been removed from Logon and related operations; instead, a HTTP status code 403 (Forbidden) with a generic reason code is returned. More detailed information is provided below. • Removed support for Container Based Processors, comprising the following API extensions: – Removed **cpb-shared-processor-usage**, **cpb-dedicated-processor-usage**, and **cpb-all-processor-usage** metrics from the CPC overview metric group. – Removed **cpb-processor-usage** metric from the Logical partitions metric group. – Removed **"cbp"** as a possible value for the **processor-type** metric in the zCPC processors metric group. – Removed the **processor-count-cbp** and **processor-count-pending-cbp** properties from the CPC object data model. – Removed the **initial-cbp-processing-weight**, **initial-cbp-processing-weight-capped**, **minimum-cbp-processing-weight**, **maximum-cbp-processing-weight**, **current-cbp-processing-weight**, **current-cbp-processing-weight-capped**, and **absolute-cbp-capping** properties from the Logical Partition object data model. – Removed the **initial-cbp-processing-weight**, **initial-cbp-processing-weight-capped**, **minimum-cbp-processing-weight**, **maximum-cbp-processing-weight**, **absolute-cbp-capping**, **number-dedicated-cbp-processors**, **number-reserved-dedicated-cbp-processors**, **number-shared-cbp-processors**, and **number-reserved-shared-cbp-processors** properties from the Image Activation Profile object data model. – Removed the **absolute-cbp-capping** and **effective-absolute-cbp-capping** properties from the Group Profile object data model. – Removed the enumeration value **"cbp"** as a possible value for the **type** field in the Capacity Record object data model. – Removed the enumeration value **"cbp"** as a possible value for the **processor-type** field in the request body for the `Add Temporary Capacity` and `Remove Temporary Capacity` operations. • For security reasons, several specific HTTP status codes and reason codes have been removed from Logon and related operations; instead, a HTTP status code 403 (Forbidden) with a generic reason code is returned. The affected operations are: `Logon`, `Establish Shared Secret Key`, and `Provide Requested MFA Information`. • There are instances when it is convenient to provide a property of a related object as a property of itself. For this reason, an ancillary property (a) qualifier was added. • Added (pc) qualifiers to several properties in the CPC object, Logical Partition object, and the Reset Activation Profile object. • Added the `Load Logical Partition from FTP` operation to the Logical Partition object. • Added the `Update Welcome Text` operation to the Console object. • Updated the `Query API Version` operation to include two new fields: **welcome-text** and **reflow-welcome-text**. • Added the `Verify Logon Password` operation to the session management services. • Added support for additional MFA types to the `Logon` operation, including the **authentication-code** field. • Added the **cpc-machine-info** property to the Console Object data model, and updated descriptions of the machine-info object properties. • Updated the `Get LPAR Controls` and `Update LPAR Controls` operations descriptions and examples. • Updated the authorization requirements for `Update Logical Partition Properties` and `Update Group Profile Properties` operations. • Changed the view-only-mode support to **true** for the **Change LPAR Controls** and **Change LPAR Group Controls** tasks in Appendix D, “Enum values for the Task object,” on page 1441.	P30805.003	P30713.004

Table 36. Summary of updates for API version 4.1 (HMC/SE Version 2.16.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Added Console Shutting Down notification information to the notification message formats section of Chapter 4, “Asynchronous notification,” on page 77. • Added the following properties to the Image Activation Profile data model: **assigned-crypto-domains**, and **assigned-cryptos**. • Added **last-used-clear-indicator** property to the Logical Partition object data model. • Added the following properties to the Console object data model: **shutdown-in-process**, **shutdown-delay-allowed**, **shutdown-delay-remaining**, **shutdown-delay-apps**, and **shutdown-delay-disable-reasons**. • The Adapter object, which was previously in DPM mode CPCs only, now has limited API support when attached to non-DPM CPCs. As a result, the following items have been added: – Two new operations are available for adapters attached to DPM mode CPCs and non-DPM CPCs: `List Permitted Adapters` and `Update Adapter Firmware` – The following adapter types were added to the **type** property for the Adapter object data model: **"osc"** (OSA Integrated Console Controller), **"ose"** (OSE for non-QDIO), **"roc2"** (RDMA over Converged Ethernet version 2), **"cl5"** (long range coupling), **"cs5"** (short range coupling), **"icp"** (Internal Coupling Link), **"hyl"** (zHyperLink Express), and **"ism"** (Internal Shared Memory). – The following values were added to the **adapter-family** property for the Adapter object data model: **"coupling"** (a coupling card), **"ism"** (an Internal Shared Memory card), **"zhyperlink"** (a zHyperLink Express), and **"not-defined"** (to indicate the **adapter-family** is not defined). – Added HTTP status code 404 with reason code 4 to the following operations: `Get Adapter Properties`, `Update Adapter Properties`, `Create Hipersocket`, `Get Storage Port Properties`, and `Change Adapter Type`. – Updated the description for reason code 4 as a possible HTTP status code 404 (Not Found) for the following operations: `Change Crypto Type`, `Delete Hipersocket`, `Get Partitions Assigned to Adapter`, `Get Network Port Properties`, `Update Network Port Properties`, and `Update Storage Port Properties`. – Added HTTP status code 409 with reason code 5 to the `List Adapters of a CPC` operation. • To enable the ability to query Hardware Management Appliance (HMA) information, the **hma-info** property was added to the Console object. The following nested objects were also added: hma-info, hma-peer-hmc-info, hma-guest-info, and hma-guest-info type-specific properties when the type value is **"se"**.		

Table 36. Summary of updates for API version 4.1 (HMC/SE Version 2.16.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• Support has been added for Server-Sent Events (SSE) over HTTP for unidirectional data flow where a client subscribes to events and receives a streamed response as notifications are emitted: – Two new primitive data types were added: String/Object URI Pattern, and String/Element URI Pattern. – HTTP/2 protocol standard is now supported. – Added reason code 7 as a possible reason code for HTTP status code 404 (Not Found) in the Common request validation reason codes table. – Added reason code 5 as a possible reason code for HTTP status code 503 (Service Unavailable) in the Common request processing reason codes table. – A new section on Server-Sent Events (SSE) was added to Chapter 4, "Asynchronous notification," on page 77. – Four new session management services operations were added in Chapter 7, "General API services," on page 111: `Create Server-Sent Events Stream`, `Update Server-Sent Events Stream`, `Delete Server-Sent Events Stream`, and `Open Server-Sent Events Stream`. • The following class specific additional properties were added to the Partition object's data model: **cpc-name** and **se-version**. • The class specific additional property **has-hardware-messages** was added to the Console object's data model. • Two class specific additional properties were added to the Logical Partition object's data model: **cpc-name** and **se-version**. • Two new operations were added to the Console object to support HMC Mobile hardware messages notifications: `Get Console Notification Preferences for Device` and `Update Console Notification Preferences for Device`. • Added the **action-settings-manage-remote-firmware-updates** field to the request body of the `Set Mobile App Preferences` operation. • Added the **action-settings-manage-remote-firmware-updates** property to the mobile-app-preferences nested object in the Console object's data model. • For the Console object and the CPC object, the descriptions of the Remote Firmware Update element object properties **service-contact-name**, **service-contact-telephone-number**, and **service-contact-email-address** were corrected to indicate that their values may be empty. • New single step operations were added. For the Console object, the `Console Single Step Install` operation was added and for the CPC object the `CPC Single Step Install` operation was added. Also, four new objects were added to the **ec-mcl-description** shared nested object: **lic-control-level**, **driver-level**, **bundle-level**, and **arom-info**. • Removed the `List Managed Virtual Machines of a Logical Partition` operation.		

Table 36. Summary of updates for API version 4.1 (HMC/SE Version 2.16.0) (continued)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.16.0 but primarily apply only to managed systems with SE version 2.16.0 or later: • Updated the `Activate Logical Partition` operation to optionally load the Logical Partition object during activation. This added several new optional fields to the request body and additional authorization requirements and status/reason codes. • Added the **speed-boost** and **ziip-boost** properties to the Logical Partition object. Each of these properties contains a boost-info nested object which contains the following properties: **boost-type**, **total-rpb-time**, and **remaining-rpb-time**. • Added the following properties to the CPC object: **software-model-purchased**, **msu-purchased**, **processor-count-permanent-ifl**, **processor-count-unassigned-ifl**, **processor-count-permanent-icf**, **processor-count-unassigned-icf**, **processor-count-permanent-iip**, **processor-count-unassigned-iip**, **processor-count-permanent-service-assist**, and **processor-count-unassigned-service-assist**. • Added the following String Enums to the **record-type** property of the Capacity Records object: – **"z-flexible-capacity"** - IBM zSystems Flexible Capacity Disaster Recovery – **"z-flexible-capacity-oocod"** - IBM zSystems Flexible Capacity On/Off Capacity on Demand record – **"tfp-hw"** - Tailored Fit Pricing Hardware. • Added the Partition Link object to support the configuration of SMC-Dv2 partition links for DPM-enabled systems. The following operations have been added for this support: – `Create Partition Link` – `Delete Partition Link` – `Get Partition Link Properties` – `List Partition Links` – `Modify Partition Link` In order to support partition links, the following additional changes were made: – Three new properties were added to the CPC object: **maximum-ism-vchids**, **minimum-fid-number**, and **maximum-fid-number**. – For the CPC object, the cpc-feature-info object nested properties, the **name** field has an additional value: **"dpm-smcd-partition-link-management"**. – For the `List Adapters of a CPC`, `List Partitions of a CPC` and `List Virtual Switches of a CPC` operations: a new query parameter, **additional-properties**, was added. – For the Partition object, the **partition-link-uris** field was added, and for partition-feature-info object nested properties, the **name** field has an additional value: **"dpm-smcd-partition-link-management"**. – For the `Create NIC` and `Update NIC Properties` operations, added reason code 557 to error code 409. – Added the **properties** query parameter to the `Get Partition Properties` operation. – For the `Get Partitions for a Storage Group` and `Get Partitions for a Tape Link` operations, updated the description of the **object-uri** field in the partition-info object nested objects table to specify the property will be null when the current user has no object access permission to the partition. – For the `Get Inventory` operation: added the **"partition-link"** class to the **"dpm-resources"** category in the description for the **resources** field.	P30805.003	P30713.004

Table 37. Summary of updates for API version 4.2 (HMC/SE Version 2.16.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.16.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 4.2. • Added the **partition-links** property to the request body contents of the `Import DPM Configuration` operation, and added information to reason code 7 of HTTP status code 400 (Bad Request) to state that the Firmware Feature enablement of the designated CPC does not support the provided configuration objects.	P30805.009 (Bundle H09)	

Table 38. Summary of updates for API version 4.10 (HMC/SE Version 2.16.0)		
Description	**HMC MCL**	**SE MCL**
The following extensions are provided by the HMC Web Services API for HMCs at version 2.16.0 and apply to all SE versions supported by the Web Services API: • Increased API version number to 4.10. • Introduced the concept of API Features. Beginning with API version 4.10 API clients must use API feature information rather than the API version to determine if specific new API functionality is available on the HMC and CPC. The feature names are listed in Table 47 on page 103. The following operations have been added for this support : – `List Console API Features` – `List CPC API Features` • Added three new properties to the Adapter object: **io-domain**, **drawer**, and **ficon-usage**. • Updated the Storage Port element object properties: – Updated the description of the **connection-endpoint-uri** property. – Updated the description of the **connection-endpoint-class** property to include the **"adapter"** value. – Added the **port-id** property. • Added reason codes 502, 503, and 504 as possible HTTP status code 409 (Conflict) error conditions reported by the `Update Storage Port Properties` operation. • Added two new properties to the CPC object: **is-host-import-key-installed**, and **primary-host-import-key-id-pattern**. • Added two new operations to the CPC object: `Import Secure Execution Key` and `Delete Secure Execution Key`. • A new Certificate Object was added, which contains the following operations: `Delete Certificate`, `Get Certificate Properties`, `Get Encoded Certificate`, `List Certificates`, and `Update Certificate Properties`. • A new operation was added to the CPC object: `Import CPC Certificate`. • Changes that were made to the Image activation profiles include a new **assigned-certificate-uris** property containing the URIs of the certificates assigned to the image activation profile, new operations to assign/unassign a certificate to/from an Image activation profile (`Assign Certificate to Image Activation Profile` and `Unassign Certificate from Image Activation Profile`), and a new **additional-properties** query parameter on the `List Image Activation Profiles` operation. **Note:** Image activation profiles named "DEFAULT" or whose name begins with "0D0" do not have any certificates in them and cannot be assigned certificates. These are special image activation profiles that are firmware created and owned. • Changes that were made to the Logical Partition object include a new property **assigned-certificate-uris** containing the URIs of the certificates assigned to the logical partition, new operations to assign/unassign a certificate to/from a logical partition (`Assign Certificate to Logical Partition` and `Unassign Certificate from Logical Partition`, and a new **additional-properties** query parameter on the `List Permitted Logical Partitions` operation. • Corrected the `Open Server-Sent Events Stream` operation URI to be GET, not POST. • Added new operations to support problem reporting: `Report a Console Problem`, `Report a CPC Problem`, `Report a Logical Partition Problem`, and `Report a Partition Problem`. • For the `Query API Version` operation, a new vendor value was added: **"n"** - a numeric vendor value, where *n* is an integer between 0 and 255. • BCPii support was added to include the support of the following operations: – Adapter Object: support the `List Permitted Adapters` operation on the non-DPM Support Element. Also added support for the MAC address and port number for this operation. – Console Object: support added for the Alternate SE's MAC address. • Added enhanced notifications for the IBM HMC Mobile app, comprising the following updates: – General API Services: Added the `Get Server-Sent Events Stream Last Event ID` operation. – Console object: Added the **enhanced-notifications-enabled** field.	P30805.014 (Bundle H14)	P30713.014 (Bundle S19)

Table 38. Summary of updates for API version 4.10 (HMC/SE Version 2.16.0) (continued)		
Description	**HMC MCL**	**SE MCL**
• A new single-load operation, `Load`, was added. **Note:** This is the recommended operation for all operating system and dump program loads. For the pre-existing profiles and activate operations that deal with **load-type**/**ipl-type** already, new Enums were added, as well as support for the new load fields and options: – `Activate Logical Partition` operation: **disk-partition-id-automatic**, **boot-record-location-use-volume-label**, **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record**. – Image Activation Profile data model: **disk-partition-id-automatic**, **last-used-boot-record-location-cylinder**, **last-used-boot-record-location-head**, **last-used-boot-record-location-record**, and **last-used-boot-record-location-volume-label**. – Load Activation Profile data model: **disk-partition-id-automatic**, **last-used-boot-record-location-cylinder**, **last-used-boot-record-location-head**, **last-used-boot-record-location-record**, and **last-used-boot-record-location-volume-label**. – Logical Partition data model: **last-used-boot-record-location-cylinder**, **last-used-boot-record-location-head**, **last-used-boot-record-location-record**, **last-used-boot-record-location-volume-label**, **last-used-device-type**, **last-used-load-program-type**, **last-used-load-operation-type**, and **last-used-disk-partition-id-automatic**. No updates were made to `Load Logical Partition`, `SCSI Load`, `SCSI Dump`, `NVMe Load`, or `NVMe Dump`. These operations will not support any of the new functionality.		

Table 39. Summary of features for API version 4.10 (HMC/SE Version 2.16.0)
Name
adapter-network-information
bcpii-notifications
cpc-delete-retrieved-internal-code
cpc-install-and-activate
create-delete-activation-profiles
dpm-ctc-partition-link-management
dpm-hipersockets-partition-link-management
dpm-smcd-partition-link-management
environmental-metrics
hmc-delete-retrieved-internal-code
ldap-direct-authentication
mobile-enhanced-push
oem-hmc-ids
pmg-child-management-permission
rc-409-15
rcl-history
rcl-progress
remote-firmware-update-rc404-4
report-a-problem

Table 39. Summary of features for API version 4.10 (HMC/SE Version 2.16.0) (continued)
Name
secure-boot-with-certificates
secure-execution-key-management
switch-support-elements

See “API features” on page 103 for a description of each new feature.

Chapter 2. Base definitions

This chapter provides basic definitions of data types, representation formats and other fundamental syntactic elements that apply across the Web Services API.

Data types

The following data types are used in the definition of the management data model, input and output parameters and notification message formats in the Web Services API.

Table 40. Primitive data types

Data type	Description
Boolean	A logical truth value: either the value **true** or the value **false**.
Byte	An integer value in the range -2^7 to (2^7)-1 (the range of a signed 8-bit integer)
Float	An IEEE 754 floating point number in the range +/-4.9E-324 to +/-3.4028235E+38. Note that, although IEEE 754 provides for representations of positive or negative Infinity and NaN, such values are not used within the API.
Long	An integer value in the range -2^63 to (2^63)-1 (the range of a signed 64-bit integer)
Integer	An integer value in the range -2^31 to (2^31)-1 (the range of a signed 32-bit integer)
Short	An integer value in the range -2^15 to (2^15)-1 (the range of a signed 16-bit integer)
String	A sequence of Unicode characters. When the number of characters in the string is bounded, the length or length range is provided in parenthesis, for example String (16) for a 16 character string, or String (0-256) for a string that may range in length from 0 (empty) to 256 characters.
String Enum	A String enumeration, i.e. a String for which the possible values are constrained to be one of a specified set of choices.
String/ URI	A String that contains a URI path used to designate object instances or operations within the API.
String/IPV4 Address	A String that contains an Internet Protocol Version 4 address presented in dotted-decimal notation. Example: `"127.0.0.1"`
String/IPV6 Address	A String that contains an Internet Protocol Version 6 address presented in colon-separated-hexadecimal notation. Leading and consecutive groups of zeros may be omitted in the representation as is conventional for IPV6 addresses presented in this form. Example: `"2001:db8:85a3::8a2e:370:7334"`

© Copyright IBM Corp. 2022, 2024

Table 40. Primitive data types (continued)	
Data type	**Description**
String/Hostname	A String that contains an internet hostname that adheres to the following standard guidelines similar to those in the Internet Engineering Task Force (IETF) RFC 1123: • length is 1-255 characters • valid characters are a-z, A-Z, 0-9, period(.), and hyphen(-) • must not begin or end with a period • must not contain consecutive periods • must not be an IPv4 address in dotted-decimal notation (see the String/IPv4 Address datatype)
String/Object URI Pattern	A String pattern to match an object or set of objects. It must be in the following form: `/api/{object-classification}/{object-id}` The pattern variables are as follows: • **object-classification** - The classification of objects to match on, as described by their **object-uri** property. A wildcard character of '*' will match any object classification. • **object-id** - The **object-id** property of an object to match on. A wildcard character of '*' will match any object. Any pattern that ends with a wildcard character may omit it. For example, `/api/cpcs` and `api/cpcs/*` are both valid and will both match any CPC object.
String/Element URI Pattern	A String pattern to match an element or set of elements. It must be in the following form: `/api/{object-classification}/{object-id}/{element-classification}/{element-id}` The pattern variables are the same as Object URI Pattern with the addition of the following: • **element-classification** - The classification of element to match on, as described by their **element-uri** property. A wildcard character of '*' will match any element classification. • **element-id** - The **element-id** property of an element to match on. A wildcard character of '*' will match any element. As with Object URI Pattern, any pattern that ends with a wildcard may omit it. **Note:** There is one object in the API that requires special rules, and that is the Console object. Since there is only one, its URI is always `/api/console` without an **object-id**. It is therefore valid, only for the Console object, to omit the **object-id** pattern variable for an Element URI Pattern to match its elements.
Timestamp	A non-negative Long integer quantity where the value represents a date and time expressed as the number of milliseconds since midnight on January 1, 1970 UTC, or the special value -1 to indicate special treatment of the timestamp field.

Table 41. Compound data types	
Data type	**Description**
Array of <T>	An ordered sequence of zero or more elements each of data type <T>. An array may be empty, i.e. have no elements contained within it.

Table 41. Compound data types (continued)	
Data type	**Description**
Object	A nested data structure providing a set of fields, each field having a name, data type and value. Object types do not formally have names. However, descriptions of these nested objects will often assign reference names to allow connections to be made in the documentation between points of use and definition for a given nested object.

Input and output representation

Except for a few special cases, the operations provided by the Web Services API expect their input and provide their output using a representation known as JavaScript Object Notation, or JSON for short. The JSON representation is also used within the bodies of notification messages emitted by the API. Unless some different representation is specifically mentioned in the description of an operation or message, all operations and messages should be understood to use JSON notation.

JavaScript Object Notation (JSON) is a lightweight, text-based, language-independent data interchange format that defines a small set of formatting rules for the portable representation of structured data. JSON can represent four primitive types (strings, numbers, booleans, and the value null) and two structured types (objects and arrays) that together provide sufficient expressive power to represent the manageable resource configuration, state, inputs, and outputs that appear in this API.

A JSON string is a sequence of zero or more Unicode characters enclosed in quotes.

A JSON object is an unordered collection of zero or more name/value pairs (sometimes referred to in this document as fields or properties), where a name is a string and a `value` is a primitive type (string, number, boolean, or null), an array, or a nested object. Each name/value pair is represented in the form **"name"**: value and is separated from the next name/value pair by a comma. The collection of name/value pairs comprising the object is enclosed by left and right braces e.g. `{ ... }`).

An array is an ordered sequence of zero or more values separated from each other by commas and enclosed in left and right square brackets e.g. `[10,20,30]`). The values in the array can be primitive or structured types, i.e. arrays of objects or arrays of arrays are permitted.

The precise BNF syntax of JSON notation is not provided in this document, but can be found in the IETF information document RFC 4726, The *application/json Media Type for JavaScript Object Notation (JSON)*, July 2006. This RFC can be found in text format on the World Wide Web at:

http://www.ietf.org/rfc/rfc4627.txt

Representing API data types in JSON

The following tables define the mapping between the API data types and their corresponding representation in JSON notation.

Table 42. Primitive data types in JSON notation	
API data type	**JSON representation**
Boolean	A JSON boolean with keywords **true** and **false**
Byte, Integer, Long, Short	A JSON number with a sign and integer component, but no fraction or exponent part.
Float	A JSON number, possibly including fraction or exponent parts. On output, values with a magnitude greater than or equal to 10^-3 and less than 10^7 are representation in floating-point format with a fraction part but not exponent part (e.g. 1.7, -32.467). Values with magnitudes outside that range are represented in scientific notation with both fraction and exponent parts (e.g. -4.23E127).

Table 42. Primitive data types in JSON notation (continued)	
API data type	**JSON representation**
String, String Enum	Represented as a JSON string enclosed in quotes.
Timestamp	An unsigned JSON number with integer component, but no fraction or exponent part.

Table 43. Compound data types in JSON notation	
Data type	**Description**
Array of <T>	A JSON square-bracket-enclosed array with elements represented according to the data type <T>.
Object	A JSON curly-brace-enclosed object, with the fields/properties of the nested object represented as name/value members of the object. The name of a property/field is used directly as the name part of the JSON object member, and the value of the field/property is provided as the value part of the member.

All strings in the JSON representation (object member names, and string values) are encoded in UTF-8.

Chapter 3. Invoking API operations

The Web Services API provides an extensive set of operations that client applications can invoke to obtain information about the manageable resources of the system, to change those resources' characteristics, and to take action on them. Because the API is designed using a web services orientation, these operations are accessed by means of Hypertext Transport Protocol (HTTP) protocol messages flowing across TCP/IP network connections.

Most aspects of HTTP protocol usage required to invoke API operations or receive responses apply universally across all of the operations of the API. Rather than repeat these details in the description of each and every operation, this common information is instead provided in this chapter. The material in this chapter should be considered to apply to each and every operation of the API unless the operation-specific description indicates otherwise. Thus, the information in his chapter should be consulted in conjunction with the operation-specific descriptions elsewhere in this document when determining how to invoke a specific API operation.

While the BCPii interface does not use the actual HTTP protocol when information is transmitted to and from the Support Element internally, the same concepts and constructs are available and used. Unless otherwise specified, requirements are the same for BCPii clients as HMC web services clients.

HTTP protocol standard

The Web Services API has been designed in accordance with the HTTP version 1.1 protocol, as defined in the W3C internet standards document *RFC 2616, Hypertext Transfer Protocol – HTTP/1.1, June 1999*. This RFC can be found in HTML format on the World Wide Web at: http://www.w3.org/Protocols/rfc2616/rfc2616.html

The API requires that all clients interact using the HTTP/1.1 or HTTP/2 protocols. The API does not support clients that use HTTP/1.0.

Note: While the API does not specifically assume or exclude any particular client user agent, its use and interpretation of HTTP elements has been designed presuming that the client application interacting with the API is a programmatic web application client or HTTP-capable scripting client rather than a standard browser-based application. Additionally, the introduction of multiplexing in HTTP/2 provides support for many concurrent requests over a single connection. Clients utilizing this support should take care to limit concurrent requests to a reasonable number to avoid running into HTTP status code 503 (Service Unavailable) errors.

Connecting to the API HTTP server

When the Web Services API is enabled, the HMC API HTTP server listens for SSL-based socket connections on TCP port 6794. The minimum required SSL/TLS version is controlled by the **Customize Console Services** task. It does not accept non-SSL connections. The set of cipher suites enabled for the HMC API HTTP server is controlled by the **Certificate Management** task on the HMC. Note, the default set of cipher suites may change with updates to the HMC if one or more of the cipher suites are found to be weak or vulnerable.

The listening port for the API HTTP server is a fixed port number and is not subject to customer reconfiguration. Thus, client applications can treat this as a well-known port number rather than requiring customer input when configuring the networking parameters the client will use to connect to the HMC.

HTTP header field usage

HTTP request and response messages include elements known as headers fields (often referred to simply as headers for short) that provide request metadata. Certain headers are required or provided in all HTTP messages, while others are present in selected messages depending on content.

© Copyright IBM Corp. 2022, 2024

This section describes the use of header fields by the Web Services API.

Required request header fields

The following HTTP request headers are relevant to all request methods (GET, PUT, POST, DELETE) and are required on all API requests (except as indicated for the Logon and Query API Version operations).

HTTP header name	Rqd/Opt	Description
Host	Required[1]	Specifies the Internet host and port number of the HMC to which the request is being directed, as obtained from the original URI given by the client application. The Web Services API enforces that this header is provided as required by the HTTP protocol, but does not check or use the value of the header in any way.
X-API-Session	Required[1, 2]	An opaque string that provides a cryptographically strong identifier of the API session (known as a session id) under which this request is executed. This header is required on all requests that require authentication. The Logon operation begins a new HMC session and includes credentials identifying the HMC user for the session. Upon successful authentication, the Logon operation returns the value to be used in the **X-API-Session** header for all subsequent requests of the same session. Failure to include this header on a request requiring authentication results in status code 403 (Forbidden) with reason code 4. Specifying an invalid session id results in status code 403 (Forbidden) with reason code 5.

Notes:

[1]Not required or allowed for requests using the BCPii interface.

[2]Not required on requests to the Query API Version and Logon operations since these operations can be performed before an API session has been established.

For requests made using the HTTP PUT or POST methods, the following additional request headers are required if a request body is being provided. If an operation being requested through POST method does not require a request body, these headers can be omitted.

HTTP header name	Rqd/Opt	Description
Content-Length	Required if request body present[1]	When used in a request, specifies the length of the request body. If omitted, the request is presumed to not contain a body. The API limits the size of request bodies in order to control usage of memory resources on the HMC. Unless a different limit is specified for a particular operation, in general the largest request body accepted by the API is 64KB. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.
Content-Type	Required if request body present[1]	When used in a request, specifies the MIME media type of the request body contained in the request. This header is required if the **Content-Length** header is supplied and specifies a nonzero request body length, otherwise status code 400 (Bad Request) will result.

Note:

[1]Not required or allowed for requests using the BCPii interface.

Optional request headers

The following HTTP request headers are relevant to all request methods (GET, PUT, POST, DELETE) and may be specified on these method requests but are not required. If present, they are interpreted by the API in the indicated way.

HTTP header name	Rqd/Opt	Description
Accept	Optional[1]	Specifies the list of response MIME media types that the client application is prepared to accept for the response to the request. This header is provides for content negotiation between the client and the server in cases where the Web Services API supports multiple possible response media types for a given operation. In the current implementation, the Web Services API supports only a single response media type for each operation. For the majority of operations, that media type is JSON (**`application/json`**), but selected operations support a different media type (indicated in the descriptions of those special operations). If this header is omitted, the Web Services API responds using the (single) media type supported for the operation. If the header is included, it must allow for the single media type that the operation supports, otherwise the request will fail with HTTP status code 406 (Not Acceptable). If an operation is extended to support multiple media types, compatibility will be maintained for existing clients that request the operation without specifying an **Accept** header.
X-Audit-Id	Optional[2]	A string that provides additional client identity information that is included in all audit records created for this request, in addition to the API user's HMC login identity. This header is intended to provide improved audit logging in the case of clients that make requests on behalf of multiple upstream users while logged into the API under a single HMC login identity. Such clients should provide the identity of their upstream user in this header so that the requests of different upstream users can be distinguished in the HMC audit logs. The HMC will use up to the first 64 characters of information from this header if present, and silently ignore the remainder of the header's value if it is longer than 64 characters.
X-Client-Correlator	Optional	A string that provides diagnostic information pertaining to this request that is of significance to the client, such as a client request number or the like. The HMC will record this information in selected diagnostic trace or log data it collects so as to allow better cross-correlation of this information with similar information maintained by the client. This data supplied in this header is intended to assist in product problem determination and does not otherwise affect the operation of the API. The HMC will use up to the first 64 characters of information from this header if present, and silently ignore the remainder of the header's value if it is longer than 64 characters.
X-API-Target-Name	Optional	Specifies the name of the target for the request. The value is the name portion of the RACF profile used for z/OS permission checking for the request. The value is checked for consistency with the URI of the request and is used internally on the Support Element to route requests to other systems. Must be specified for all BCPii requests except for the /api/cpcs operation for the local system.

HTTP header name	Rqd/Opt	Description
Note: [1]Not allowed for requests using the BCPii interface. [2]For the BCPii interface, this is automatically set by BCPii and contains the user ID associated with the calling program.		

Standard response headers

The following HTTP response headers are always provided in the response to all requests.

HTTP header name	Description
Date	The date and time, from the perspective of the HMC's clock, at which the response message was generated. As required by the HTTP protocol specification, this date is an HTTP full date sent in the RFC 1123-defined fixed length format. Example: `Sun, 08 Oct 1961 10:08:00 GMT`

The following HTTP response headers are provided in the response to all requests except those that result in a 204 (No Content) HTTP status code.

HTTP header name	Description
Content-Length	When used in a response, specifies the length of the response body. If omitted, the response does not contain a body.
Content-Type[1]	When used in a response, specifies the MIME media type of the response body. This response header is provided any time the **Content-Length** header is provided and specifies a nonzero length.
[1]Not applicable to the BCPii interface.	

Additional response headers

Some operations may return additional response headers beyond those described in "Standard response headers" on page 62. The following table describes these possible additional response headers. Operations that return these additional headers indicate that they do so in the operation description.

HTTP header name	Description
Location	The URI of the resources that was created by the operation. This header is provided for operations that complete successfully with an HTTP status code of 201 (Created).
X-API-Session[1]	An opaque string that provides a cryptographically strong identifier of the API session that was created for the client. This header is provided in the response to a successful Logon operation.

HTTP header name	Description
X-Request-Id	A string that provides diagnostic information identifying the request from the perspective of the HMC. This same information is included in the API log entries that are recorded by the HMC for the request. If captured by the client from a response, a client application developer or support technician can use this information to locate the HMC API log entry corresponding to a particular request. The value of this header will be 64 characters or less. This header is provided in the responses to all requests.
[1]Not applicable to the BCPii interface.	

Media types

The following media types are applicable to the use of the Web Services API, and thus may appear in the values of **Accept** or **Content-Type** header fields.

MIME media type	Description
`application/json`	JavaScript Object Notation (JSON), as described by RFC 4627. This media type is used by the Web Services API for both request and response representation for the majority of the operations in the API. The JSON text is encoded using the UTF-8 charset.
`application/vnd.ibm-z-zmanager-metrics`[1]	Custom output format used for providing the results to the Get Metrics operation of the metrics service. The result text is encoded using the UTF-8 charset.
`application/xml`[1]	Extensible Markup Language, used for the input and output formats for the Export Performance Policy and Import Performance Policy operations of the workload object. The XML text is encoded using the UTF-8 charset.
`application/octet-stream`[1]	Binary data. This media type is used by the Web Services API for the request representation for the `Mount Virtual Media Image` operation of the Virtual Server object.
Note: [1]Media type currently not supported for BCPii interface.	

HTTP status codes

The HMC API provides standard HTTP status codes in the response to requests to indicate the success or failure of the request. Unless stated otherwise in the description of an operation, the following general interpretations of the status code values apply.

HTTP status code	Description/Causes
200 (OK)	The request has succeeded completely. A response body is provided that contains the results of the request.

HTTP status code	Description/Causes
201 (Created)	The request has succeeded completely and resulted in the creation of a new managed resource/object. The URI for the newly created managed resource is provided in a **Location** header. (POST methods only)
202 (Accepted)	The request was successfully validated and has been accepted to be carried out asynchronously.
204 (No Content)	The request succeeded completely, and no additional response information is provided.
400 (Bad Request)	The request was missing required input, had errors in the provided input, or included extraneous input. Additional information regarding the error is provided in an error response body that includes a reason code with additional information.
403 (Forbidden)	Multiple error conditions result in this status code: • The request requires authentication but no **X-API-Session** header was provided, or one was provided but the session ID was invalid. • The user under which the API request was authenticated is not authorized to perform the requested operation.
404 (Not Found)	The URI does not designate an extant resource, or designates a resource for which the API user does not have object-access permission.
405 (Method Not Allowed)	The request specifies an HTTP method that is not valid for the designated URI.
406 (Not Acceptable)	The **Accept** header for the request does not include at least one content representation supported by the Web Services API.
409 (Conflict)	The managed resource is in an incorrect state (status) for performing the requested operation. Additional information regarding the error is provided in an error response body that includes a reason code with additional information.
413 (Request Entity Too Large)	The request includes a request body that is too large. Unless a different limit is specified for a particular operation, in general the largest request body accepted by the API is 64 KB.
415 (Unsupported Media Type)	The **Content-Type** header for the request specifies a representation that is not supported by the Web Services API.
500 (Server Error)	A server error occurred during processing of the request.
501 (Not Implemented)	The request specifies an HTTP method that is not recognized by the server (for any resource). **Note:** The response body that accompanies this error is not a JSON response body as defined in "Error response bodies" on page 65.
503 (Service Unavailable)	The request could not be carried out by the HMC due to some temporary condition.
504 (Gateway Timeout)	A time out occurred trying to route a BCPii originated request to a remote system.
505 (HTTP Version Not Supported)	The request specifies an HTTP protocol version that is not supported by the Web Services API.

Error response bodies

For most 4xx and 5xx HTTP error status codes, additional diagnostic information beyond the HTTP status code is provided in the response body for the request. The API client can use the **content-type** header to determine the type of information in the response body. If the value of the **content-type** header is **application/json**, the following information is provided in the form of a JSON object containing the fields in the following table.

Note: The content of the error response bodies will vary when the error was detected by z/OS BCPii.

Field name	Type	Description
request-method	String	The HTTP method (DELETE, GET, POST, PUT) that caused this error response.
request-uri	String	The URI that caused this error response.
request-query-parms	Array of query-parm-info objects	An array of query-parm-info objects (described in the table below) that identify the query parameters specified on the request and their values. Each query-parm-info object identifies a single query parameter by its name and includes its value(s). If the request contains no query parameters, this field is omitted.
request-headers	header-info object	A header-info object (described in the table below) that describes the HTTP headers specified on the request. If the request contains no HTTP headers, this field is omitted.
request-authenticated-as	String	The name of the HMC user associated with the API session under which the request was issued. If the request was issued without an established session or there is no HMC user bound to the session, this field is omitted.
request-body	String	The request body, in the form of a JSON document. Note that, since it is in the form of a JSON document, this may not be exactly what was submitted by the API client program, but it is semantically equivalent. If the request body could not be parsed or some other error prevented the creation of a JSON document from the request body, this field is omitted and the request body is instead available in the **request-body-as-string** field.
request-body-as-string	String	The complete request body, or some portion of the request body, exactly as it was submitted by the API client program. The **request-body-as-string-partial** field indicates whether the complete request body is provided. If the **request-body** field is present, this field is omitted.
request-body-as-string-partial	Boolean	When the **request-body-as-string** field is present, this boolean indicates whether the **request-body-as-string** field contains only part of the request body (true) or the entire request body (false). If the **request-body-as-string** field is not present, this field is omitted.
http-status	Integer	HTTP status code for the request.
reason	Integer	Numeric reason code providing more details as to the nature of the error than is provided by the HTTP status code itself. This reason code is treated as a sub-code of the HTTP status code and thus must be used in conjunction with the HTTP status code to determine the error condition. Standard reason codes that apply across the entire API are described in "Common request validation reason codes" on page 66. Additional operation-specific reason codes may also be documented in the description of the specific API operations.
message	String	Message describing the error. This message is not currently localized.
stack	String	Internal HMC diagnostic information for the error. This field is supplied only on selected 5xx HTTP status codes.

Field name	Type	Description
error-details	Object	A nested object that provides additional operation-specific error information. This field is provided by selected operations, and the format of the nested object is as described by that operation.
bcpii-error	Boolean	Indicates whether an error was encountered/detected by z/OS BCPii before sending the request to the Support Element or after receiving a response back from the Support Element (true), or was detected on the Support Element after receiving the request from BCPii (false). **Note:** This field is only returned when the BCPii interface was used for the request.

Each query-parm-info object contains the following fields:

Field name	Type	Description
{query_parm_name}	Array of Strings	The value of the *{query_parm_name}* query parameter. If this query parameter was specified multiple times, there will be multiple entries in this array, one for each instance of this query parameter.

The header-info object contains the following field(s), one for each header present on the request:

Field name	Type	Description
{header_name}	String or Array of Strings	The value of the *{header_name}* HTTP request header. It will be either a single string or an array of strings.

Usage notes:

- The message provided in the **message** field is primarily intended as a convenience for use by developers when developing and testing client applications. Because it is not localized, it may not be appropriate for client applications to simply pass this message on to their clients when reporting errors to those upstream clients. Instead, client applications can use the value in the **reason** field as a key in obtaining a client-provided message that may be more appropriate to use.
- Because the reason code is treated as a sub-code of the HTTP status code, the same reason code value is often defined for multiple different HTTP status codes and has a different meaning in each case. For example, reason code 1 when considered for a 400 (Bad Request) status code has a different meaning than when considered for a 403 (Forbidden) status code. For this reason, client applications that make decisions based on the reason codes should always include checking the HTTP status code as part of the relevant logic (e.g.test for status code == 400 AND reason code == 1, not just reason code == 1 alone).

Common request validation reason codes

The Web Services API performs request validation on each request it receives to ensure the request is correctly formed and appropriate before it begins processing the request. Many errors of basic request syntax can occur on all or a large number of the operations provided by the API. Validation for these kinds of errors is done in a common way across all of the operations and results in a common (not request-specific) reason code being reported if errors are detected. Other validation is operation-specific by nature, and results in operation-specific reason codes when errors are detected.

The following table provides the HTTP status codes and reason codes for common request validation. These status and reason codes may be reported on any of the operations of the API.

HTTP status code	Reason code	Description
400 (Bad Request)	1	The request included an unrecognized or unsupported query parameter.
	2	A required request header is missing or invalid.
	3	A required request body is missing.
	4	A request body was specified when not expected.
	5	A required request body field is missing.
	6	The request body contains an unrecognized field (i.e. one that is not listed as either required or optional in the specification for the request body format for the operation).
	7	The data type of a field in the request body is not as expected, or its value is not in the range permitted.
	8	The value of a field does not provide a unique value for the corresponding data model property as required.
	9	The request body is not a well-formed JSON document.
	11	The length of the supplied request body does not match the value specified in the **Content-Length** header.
	13	The maximum number of logged in user sessions for this user ID has been reached; no more are allowed.
	14	Query parameters on the request are malformed or specify a value that is invalid for this operation. Common causes include the inability to successfully decode a parameter element, the presented parameters are not in the expected key=value format, the value is not a valid regular expression, a required parameter is missing, multiple instances of a parameter are present on an operation that does not permit multiple instances of that parameter, or the value is not a valid enum for the operation.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body. A prerequisite condition or dependency among request body fields is not met.
	18	The request body contains a field whose presence or value is inconsistent with the type of the object. Such a requirement is often described in an object's data model as the field having a prerequisite condition on a "type", "family", or similar property that identifies an object as being of a particular type. Such a property is typically, but not necessarily, immutable.
	19	The request body contains a field whose corresponding data model property is no longer writable. In certain earlier HMC and/or SE versions the property is writable, but later versions do not allow changing the property through the Web Services APIs. This could be due to a change in the underlying system-management model, or the property may have become obsolete.
	20	The request body contains a field or value that is directly or indirectly dependent on the version of the SE that is targeted by or associated with the request operation, and that SE is not at a version that supports or provides the field or value.
	21	There was an error decoding either the URI, method, header, or body for a request received over the BCPii interface.

HTTP status code	Reason code	Description
	4000[1]	A non-null address was provided for the request body data area but the length passed in for the area was 0.
	4001[1]	The length provided for the request body data area exceeded the supported 64KB.
	4002[1]	The request body encoding is not the expected UTF-8.
	4003[1]	The request body encoding is not the expected IBM-1047.
	4004[1]	The request body contains malformed JSON. See the message, in the error response body, for further information.
	4005[1]	The address provided for the required URI data area is null.
	4006[1]	A non-null address was provided for the URI data area but the length passed in for the area was 0.
	4007[1]	The length provided for the URI data area exceeded the supported 2048 bytes.
	4008[1]	The contents of the URI data area evaluated to an empty string. A valid, non-empty string, is required for the URI.
	4009[1]	A non-null address was provided for the target name data area but the length passed in for the area was 0.
	4010[1]	The length provided for the target name data area exceeded the supported 256 bytes.
	4011[1]	A non-null address was provided for the client correlator data area but the length passed in for the data area was 0.
	4012[1]	The length provided for the client correlator exceeded the supported 64 bytes.
	4013[1]	A null address was provided for the required response body data area. A valid address to a data area, minimum 500 bytes, is required.
	4014[1]	A non-null address was provided for the response body data area but the length of the data area did not meet the minimum 500 bytes requirement.
	4015[1]	The encoding parameter contained an unrecognized value. See the IDF corresponding to the programming language for valid encoding constants.
	4016[1]	The HTTP method parameter contained an unrecognized value. See the IDF corresponding to the programming language for valid HTTP method constants.
	4017[1]	The length of the response date data area is not the required 29 bytes.
	4018[1]	The length of the request ID data area is not the required 64 bytes.
	4019[1]	The length of the location data area is not the required 2048 bytes.
	4020[1]	The URI contains an unrecognized value.
	4021[1]	The target name contains an unrecognized value.
	4022[1]	The data area associated with the URI is not accessible by BCPii.

HTTP status code	Reason code	Description
	4023[1]	The data area associated with the request body is not accessible by BCPii.
	4024[1]	The data area associated with the target name is not accessible by BCPii.
	4025[1]	The data area associated with the client correlator is not accessible by BCPii.
	4026[1]	The data area associated with the response date is not accessible by BCPii.
	4027[1]	The data area associated with the request ID is not accessible by BCPii.
	4028[1]	The data area associated with the location is not accessible by BCPii.
	4029[1]	The data area associated with the response body is not accessible by BCPii.
	4030[1]	The data area associated with the request parameter is not accessible by BCPii.
	4031[1]	The calling program resides in an unsupported environment.
	4032[1]	The calling program is disabled.
	4033[1]	The calling program is holding one or more locks.
	4034[1]	The calling program is not in task mode.
	4035[1]	The request is not supported for this type of REXX environment.
	4036[1]	A valid target name is required for this type of request, however the target name value was either not provided or evaluated to an empty string.
	4300[1]	The calling REXX program did not provide the required URI.
	4301[1]	The calling REXX program did not provide the required HTTP method.
	4302[1]	The calling REXX program provided a URI that was not a valid string.
	4303[1]	The calling REXX program provided a bad HTTP method value.
	4304[1]	The calling REXX program provided a target name that was not a valid string.
	4305[1]	The calling REXX program provided a client correlator that was not a valid string.
	4306[1]	The calling REXX program provided a request body that was not a valid string.
	4307[1]	The calling REXX program provided a bad encoding value.
	4308[1]	The calling REXX program provided a timeout value that was not a valid integer.

HTTP status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
	4	The request requires authentication but no X-API-Session header was specified in the request.
	5	An X-API-Session header was provided but the session id specified in that header is not valid.
	301	The operation cannot be performed because it targets a CPC that does not support Web Services API operations.
	4000[1]	The user ID associated with the calling program is either lacking authority to use BCPii or to the particular resource associated with the request. For details regarding authority to BCPii, refer to https://www.ibm.com/docs/en/zos/2.4.0?topic=bcpii-general-security-product-authority. For details regarding authority to issue the request, see Appendix A, "Base Control Program internal interface (BCPii)," on page 1411.
	4001[1]	The calling program is running in a problem state, PKM8-15, and is not executing from an APF-authorized library.
	4002[1]	The calling REXX program is running in a problem state, PKM8-15, and is not executing from an APF-authorized library.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. For URIs that contain object ID and/or element ID components, this reason code may be used for issues accessing the resource identified by the first (leftmost) such ID in the URI.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. For URIs that contain object ID and/or element ID components, this reason code may be used for issues accessing the resource identified by the first (leftmost) such ID in the URI.
	4	The object designated by the request URI does not support the requested operation.
	5	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. More specifically, this reason code indicates issues accessing the resource identified by the element ID component in the URI. Such an element ID is typically the second (counting left to right) ID component in the URI.
	6	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission. More specifically this reason code indicates issues accessing the resource identified by the element ID component in the URI. Such an element ID is typically the second (counting left to right) ID component in the URI.
	7	The identified Server-Sent Events stream or BCPii Registration ID was not found. [Added by feature **bcpii-notifications**]

HTTP status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	3	The operation cannot be performed because the object designated by the request URI is currently locked to prevent disruptive changes from being made.
	4	The operation cannot be performed because the CPC designated by the request URI is currently enabled for DPM.
	5	The operation cannot be performed because the CPC designated by the request URI is currently not enabled for DPM.
	6	The operation cannot be performed because the object hosting the object designated by the request URI is not in the correct state.
	8	The operation cannot be performed because the request would result in the object being placed into a state that is inconsistent with its data model or other requirements. The request body contains a field whose presence or value is inconsistent with the current state of the object or some aspect of the system, and thus a prerequisite condition or dependency would no longer be met.
	9	The operation cannot be completed because it is attempting to update an effective property when the object is not in a state in which effective properties are applicable. More specifically, the request body contains one or more fields which correspond to a property marked with the (e) qualifier in the data model, and the object's **effective-properties-apply** property is **false**.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	11	The operation cannot be performed because it requires a fully authenticated session and the API session that issued it is only partially authenticated.
	12	The operation cannot be performed because a firmware feature that prohibits the operation is currently enabled. The **error-details** field of the response body contains an error-feature-info object identifying the firmware feature whose current enablement status is invalid for the operation. The error-feature-info object is described in the next table.
	13	The operation cannot be performed because a firmware feature required by the operation is currently disabled. The **error-details** field of the response body contains an error-feature-info object identifying the firmware feature whose current enablement status is invalid for the operation. The error-feature-info object is described in the next table.
	14	The operation cannot be performed because the data that would be changed is currently subject to Data Replication from a PRIMARY or PEER HMC. When the HMC is a Data Replication REPLICA any data that is subject to replication cannot be changed.
	15[1]	The response content cannot be returned to the BCPii client application because the data area provided for the response body is not large enough to contain it. [Added by feature **rc-409-15**]

HTTP status code	Reason code	Description
[1]This reason code is only applicable to operations initiated through the BCPii interface.		

The error-feature-info object contains the following fields:

Table 44. error-feature-info object properties

Field name	Type	Description
scope	String Enum	The scope of the firmware feature requirement. The valid values are: • **"cpc"** - The firmware feature enablement requirement applies to the CPC object involved in the operation. • **"partition"** - The firmware feature enablement requirement applies to the Partition object involved in the operation.
name	String Enum	The name of the firmware feature requirement. • When the **scope** is **"cpc"**, this is a firmware feature name from the CPC's **available-features-list** property. • When the **scope** is **"partition"**, this is a firmware feature name from the Partition's **available-features-list** property.

Common request processing reason codes

Certain common error conditions can be encountered during the processing of many of the operations of the API. When they are encountered they are reported using the same HTTP status and reason code by any operation of the API that may encounter them.

These common request processing reason codes are listed in the following table:

HTTP status code	Reason code	Description
500 (Server Error)	0 - 39	An internal processing error has occurred and no additional details are documented.
	4119[1]	The BCPii transport rejected the particular request. Activate CTRACE with CTRACE option "ALL" and reissue the request. If the request fails again, turn off CTRACE, collect the SVCDUMP, and contact your next level of support.
	4130[1]	The support element failed to return the required information needed for BCPii address space to come up. Action: 1. If this error occurs during BCPii initialization, restart BCPii manually (S HWISTART). 2. If restarting BCPii manually fails, perform the following steps to re-drive the SE recovery process to return the required information: a. Issue the command VARY CN(*),ACTIVATE from Operating System Messages b. Issue a command (any command) from Operating System Messages c. Manually restart BCPII (S HWISTART) 3. If the above suggested actions still fail, IPL is required to restart BCPii.
	4208[1]	The support element rejected the particular request. This could occur for any number of reasons including: the SE is busy, the SE is rebooting, etc. Consider retrying the request one or more times. If the problem persists, activate CTRACE with CTRACE option "ALL" and reissue the request. Then turn off CTRACE, collect the SVCDUMP, and contact your next level of support.
	4209-4211[1]	The SEND BCPii permission was not granted to The LPAR on this support element, contact your next level of support.
	4212[1]	The support element rejected communication from BCPii, likely for one of the following reasons: the SEND BCPii permission was not granted to the LPAR on this support element or the SE/HMC is missing the required MCL. Contact your next level of support.
	4313[1]	C/ASM applications: the length of the data area provided for the response body is not large enough to contain the response from the SE. Increase the data area size to the recommended amount and re-issue the request. REXX applications: the size of the response exceeds the support 2.5 MB response body limit for this environment, consider using C or ASM to re-issue the request with the recommended data area size.
	4xxx[1]	An internal error occurred processing the request, contact your next level of support.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.
	3	This request would exceed the limit on the number of concurrent API requests allowed.
	4	The limit of concurrent requests has been reached for a specific BCPii source partition.
	5	This request would exceed the limit on the number of concurrent Server-Sent Events streams allowed for a session.

HTTP status code	Reason code	Description
504 (Gateway Timeout)	0	The address for the specified target was unable to be determined.
	1	The targeted API server was unable to be reached.
	2	The targeted API server was reached and a request was successfully submitted but the response did not arrive in a timely enough manner to satisfy the time out specified by the client.
	3	The targeted API server was reached and a request was successfully submitted but the response did not arrive in a reasonable amount of time.
	4224[1]	No response was received from the support element, after waiting the amount of time specified by the timeout request parameter, and BCPii timed out the request. Contact your next level of support to check if connectivity to the support element is still there.
[1]This reason code is only applicable to operations initiated through the BCPii interface.		

Use of chunked response encoding

For most API operations, the size of the response data is modest and therefore standard HTTP response payload transfer encoding is used. In this encoding, the length of the entire payload of the response message is provided in the response before any of the contents of the response payload are written to the socket connection. But some operations, such as the Get Inventory operation of the Inventory service and the Get Metrics operation of the Metrics service, can produce very large responses. Use of standard transfer encoding for these kinds of operations is inefficient for the HMC because it requires the entire response be generated and buffered before any of it is sent in order to compute and send the total length of the response body before sending any of the contents of the response data.

To avoid the need for the buffering the entire response, and to instead allow the response to be transmitted in smaller segments as they are prepared, operations that return large responses use HTTP chunked response encoding instead. Chunked transfer encoding is an HTTP V1.1 data transfer feature that allows the payload of the response message to be split into a sequence of smaller parts known as chunks, with the size of each chunk transmitted as part of the chunk rather than requiring the transmission of the size of the full response payload.

Chunked transfer encoding is defined in the HTTP/1.1 protocol standard, RFC 2616, cited earlier in this section.

The HTTP protocol standard requires that all HTTP/1.1 applications (client or server) be capable of receiving and handling chunked transfer-encoded messages, so the use of this encoding by the API HTTP server is within the options allowed by the protocol standard. However, since this format may be unexpected to naively-written applications, its use is limited by the API HTTP server to the special circumstances that warrant its use to improve performance or efficiency. Therefore, a client application can safely assume that an operation will not use chunked transfer encoding for its responses unless the use of this encoding is specifically mentioned in the description of the operation.

The data returned from the Support Element to z/OS BCPii will never use chunked encoding. The Support Element will "de-chunk" any response bodies that use chunked encoding and encode it using the specified character set before sending the data to z/OS BCPii. Since the HWIREST interface requires the application to provide a pre-allocated buffer to hold the response data and the Support Element knows this buffer size, no response data will ever be sent if the response data is too large for the buffer. Instead a specific error is returned to z/OS BCPii indicating the application buffer is too small.

Filter query parameters

Some operations allow for the (optional) use of designated query parameters for conveying additional request parameters. Although query parameters can be used to convey various kinds of additional

request information, most operations that make use of query parameters do so for the purpose of filtering the response entries to a subset of what would otherwise be returned. For example, this kind of filtering is typically provided on operations that are described as List operations (e.g. `List Logical Partitions of a CPC`). This section describes the interpretation/handling of filter-type query parameters across all of the operations of the API.

As would be expected, if an operation is invoked without specifying any of its possible filter-type query parameters, the operation returns all of the result entries applicable to the request. For example, the `List Logical Partitions of a CPC` operation invoked with no filtering query parameters returns all of the Logical Partition objects in the CPC to which the API user has access.

If one or more filter-type query parameters are specified, the combination of those parameters specifies a logical match expression that is evaluated against each entry that is a candidate for inclusion in the result to determine if the entry is included or not. Within that expression, there may be multiple occurrences of the same-named query parameter and/or there may be occurrences of differently-named query parameters. The query parameters are interpreted as a logical expression using the following rules:

- Multiple occurrences of the same-named query parameter are interpreted as a group that is connected by a logical OR operation among all of query parameters with the same name. An entry remains a candidate for inclusion in the result as long as it matches at least one of the values specified for this particular query parameter.
- Occurrences of differently-named query parameters are first organized into OR'ed groups as mentioned above, and then these groups are interpreted as being connected by logical AND operations. Thus an entry is included in the result only if it matches at least one value from each of the differently-named groups of parameters.
- As an example, a query string of "name=fee&type=fie&name=foe&type=fum" is interpreted as specifying the expression (name=fee OR name=foe) AND (type=fie OR type=fum). Note that the order in which the query parameters appear in the string is not important.

As a filter-type query parameter is applied against a candidate entry, it is determined to match or not as follows:

- If the query parameter is of data type String, the parameter's value is interpreted as a regular expression pattern and is considered to match if the corresponding String property of the candidate entry matches the pattern.
- If the query parameter is of data type String Enum, the parameter's value is compared against the corresponding Enum property of the candidate entry and is considered to match if they are exactly the same value.
- If the query parameter is of data type Boolean, the parameter's value is compared against the corresponding Boolean property of the candidate entry and is considered to match if they are both true or both false.
- If the query parameter is of data type String/ URI, the parameter's value is interpreted as a canonical URI path and is considered to match if the corresponding property of the candidate entry is exactly the same value.

Regular expression syntax

The values of String-type filtering query parameters are interpreted as regular expressions. The regular expression syntax used is the same as that used by the Java programming language, as specified for the java.util.regex.Pattern class. Documentation on that syntax can be found at on the following web page: http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html

Chapter 4. Asynchronous notification

The Web Services API includes an asynchronous notification facility by which client applications may subscribe to and receive notification messages regarding a set of predefined management events. These events include:

- Addition and removal of managed objects to/from the inventory of resources that are managed by the HMC.
- Changes to specified properties of managed object instances.
- Changes to the operational status of managed objects.
- Completion of asynchronously processed jobs.
- Addition of entries to the HMC's audit log.
- Addition of entries to the HMC's security log.
- Addition of entries to the HMC's event log.
- New and refreshed operating system messages.

The HMC notification facility provides two services for subscribing to notifications. One is based on the Java™ Message Service (JMS) architecture and API for exchanging messages among two or more applications. The other is using Server-Sent Events (SSE) over HTTP for unidirectional data flow where a client subscribes to events and receives a streamed response as notifications are emitted.

Refer to "Asynchronous notification support" on page 1428 for details on asynchronous notification support for the BCPii interface. [Added by feature **bcpii-notifications**]

Grouping of notifications

A particular managed resource may often experience a series of changes that occur in rapid succession. This might occur, for example, when a user uses an object's Details task in the HMC UI to make a set of changes to the object's properties and then selects the Finish button to complete the task. All of the pending property changes collected by the task will be made on the managed object in very quick succession in response to the Finish button being selected, rather than before as the user was interacting with the task.

In order to reduce the notification traffic in these situations, the notification messages have been designed to allow the HMC to report multiple changes of the same type (e.g. property changes, status changes) for the same managed resource in a single message rather than generating a distinct message for each change. In order to do such grouping, the HMC may delay generation of a notification message for a change for a brief period of time in order to allow coalescing of that change report with others that occur for the same managed resource within the coalescing time window. This optimization will be performed while maintaining the following characteristics:

- Grouping of notifications may be done for property change, status change, operating system message, and log entry notifications, but will not be done for other notification types.
- Notifications will be buffered for a maximum of 1 second.
- The grouping of change reports will not obscure a client's ability to correctly observe the temporal ordering of the individual changes made to a particular object or between objects based on the messages sent to a session. That is, notification messages will always be generated so that the ordering of the messages as determined by their sequence number identifiers reflects the temporal order in which the changes were actually made. All of the changes reported to a session in a message with a lower sequence number will have occurred before any of the changes reported in a message with a higher sequence number. Further, the ordering of change reports within a particular message reflects the order in which they occurred to that object as well.
- Coalescing of multiple changes into a single notification message will occur for at most a single pending notification message (thus, of a single type) at a time. If a need arises to report a change of a different

© Copyright IBM Corp. 2022, 2024

type than is currently pending (for example, a need to report a status change when there is currently a set of pending property change reports), coalescing will end for that pending message and it will be posted to JMS topics and SSE streams as appropriate. This is necessary in order to maintain the API client's ability to correctly observe temporal ordering.

- The grouping of change reports into notification messages occurs independently for each session, so that one session may receive a different distribution of change reports across notification messages than another session.

The degree to which message grouping occurs or not depends on the timing of changes and possibly other considerations and thus is to be strictly considered an optimization and not guaranteed behavior. Client applications should infer no particular semantic significance to change reports being delivered in a single message vs. a sequence of messages.

Java Message Service (JMS) basics

In the JMS model, message-based communication between producing and consuming applications is coordinated by an intermediate component known as a message broker that acts as the clearinghouse for message exchange. The message broker provides a registry of the available destinations to which messages can be posted, and a store for messages that have been posted but not yet consumed.

Applications acting in the role of message producer create messages and post them (through the broker) to the destination appropriate for the type of message. The messages are associated with the destination and retained by the broker until they have been consumed by interested applications.

Applications acting in the role of message consumer connect to a message destination (again, through the broker) in order to express interest in receiving messages posted to it. As messages are posted to the destination by producers, the broker makes the messages available to interested consumers which then receive and process the message.

In the Web Services API notification facility, the HMC acts both as the message broker and the message producer for API notification messages. API client applications act as message consumers.

For the broker function, the HMC includes an integrated JMS message broker implementation based on Apache ActiveMQ, an open, standards-based implementation of JMS. This integrated broker is configured to allow internal HMC function to act as message producers, and to allow external applications to connect as message consumers. However, external applications cannot produce and send messages using the HMC integrated broker.

Connecting to the API message broker

As part of the Web Services API, the HMC provides an integrated JMS message broker based on Apache ActiveMQ. This message broker is active on the HMC whenever the Web Services API is enabled.

When active, the integrated broker listens for client connections using the following transports supported by ActiveMQ:

- OpenWire flowing over SSL connections, listening port: 61617.
- STOMP (Streaming Text Oriented Messaging Protocol) flowing over SSL connections, listening port: 61612.

The minimum required SSL/TLS version by the broker on these ports is controlled by the **Customize Console Services** task.

The listening ports for the message broker are fixed port numbers and are not subject to customer reconfiguration. Thus, client applications can treat them as well-known port numbers rather than requiring customer input when configuring the networking parameters the client will use to connect to the HMC.

In order to connect to the integrated message broker, clients must provide certain authentication information for the HMC user making the connection. If the HMC user is configured to use multi-factor authentication, then the client must provide information about the HMC user's API session that is to be associated with the message broker connection. The user name field in the connection request must

contain the API session ID, and the password field must contain the session-specific authentication credential. Both of these are available in the response body of the `Logon` operation that created the API session: the **api-session** and **session-credential** fields, respectively.

If the HMC user is not configured to use multi-factor authentication, then the client may provide the authentication information as described immediately above, or it may provide a valid HMC user name and logon password in order to identify the HMC user making the connection. The user name and password are validated using the standard HMC user authentication mechanisms before allowing the connection to succeed.

The integrated message broker does not allow any anonymous or unauthenticated connections.

Per-session notification topics

As part of its access control enforcement, the Web Services API limits the distribution of notification messages to clients that have object-access permission to the managed object for which the notification was generated.

In order to accomplish this, the API does not define a single (or fixed number of) notification topics to which all messages are posted and from which any or all clients can receive messages. Rather, the API uses per-session notification topics.

In this approach, each API session is associated with a set of JMS destinations that are created by the HMC when the session is created or other actions are performed using the session, and are used for providing notifications destined to that session. The names of the object notification and job completion destinations are returned as part of the results from the Logon operation. The names of all destinations available to a session are returned by the `Get Notification Topics` operation. Each session has the following per-session notification destinations:

- An object notification topic, used by the HMC to send notifications that pertain to the inventory and status of managed objects that this session has permission to access.
- A job notification topic, used by the HMC to send notifications that pertain to the status of asynchronous operations that are initiated by this session.
- An audit notification topic, used by the HMC to send notifications that pertain to the HMC's audit log, but only if the client has permission to the **Audit and Log Management** task. Without the required task permission, there is no such destination available to the session.
- A security notification topic, used by the HMC to send notifications that pertain to the HMC's security log, but only if the client has permission to the **View Security Logs** task. Without the required task permission, there is no such destination available to the session.
- A console event notification topic, used by the HMC to send notifications that pertain to the HMC’s event log, but only if the client has permission to the **View Console Events** task. Without the required task permission, there is no such destination available to the session
- An operating system message notification topic, used by the HMC to send notifications that pertain to new and refreshed operating system messages. More than one topic of this type can exist for a single API session.

  Unlike the other topic types, topics of this type are not created by default when the session is created. They are created when the user targets the `Open OS Message Channel` operation at a Logical Partition or Partition object. Operating system messages begin to flow after the user first connects to the topic. If refresh messages are desired, they are published to the topic immediately following a connection being established to the topic. New messages are sent as they are received. When there are no connections remaining to the topic, the flow of messages stops. The topic will persist, allowing for a reconnect. If the user reconnects, messages will flow as if it was the first connect. The topic is destroyed only when the user performs the Logoff operation or the session is otherwise destroyed.

The session is also associated with an HMC user (identified during API session logon) that in turn has a set of object-access permissions defined for it that determine the managed resources that it is authorized to access. The HMC user also has a set of task permissions defined for it that determine the tasks that it is authorized to perform.

As notification messages are created for managed resource changes or other events, they are distributed to sessions according to the object-access permissions of those sessions. More specifically, when a notification is generated pertaining to some managed resource, it is published to the object notification topics of all sessions for which the related API user has object-access permission to that managed resource, and is omitted from the object notification topics of sessions for which the user does not have object-access permission. As a result, a particular API session will have access to all notifications for all managed resources to which its API user has access permission, but will not have access to notifications for managed resources that it does not.

Notification messages for job completion are sent only to the job notification topic of the API session that initiated the asynchronous processing represented by the job.

Notification message formats

Several types of notification messages are provided by the API. The JMS messages created for all types of notifications share a common set of message characteristics and header fields, which are extended with additional header fields and message body formats that vary by the type of notification.

Common message characteristics

The characteristics described in this section apply to all notification messages published by the Web Services API.

Message format

The following JMS message characteristics apply to all notification messages sent by the Web Services API. These characteristics are echoed in the message themselves in the values of the standard JMS message header fields.

Characteristic	Description
Destination	The per-session object or job notification topic as indicated below for each type of notification.
Message type	Text message. The contents of the body vary by the type of notification.
Delivery mode	Nonpersistent.
Expiration	No expiration.
Priority	4 (highest normal priority)
Message ID	A unique message ID for the message.
Correlation ID	Not set by the API.
Timestamp	The time, from the HMC's perspective, that this message was sent.

In addition to the standard JMS message headers, the following additional message properties are provided in all notification messages to allow for message selection:

Message property name	Description
notification-type	The type of notification contained in this message, varies by notification type.
session-sequence-nr	The sequence number of this notification within the session. This number starts at 0 when the API session is created, and is incremented each time a notification message is published to this session.

Message property name	Description
global-sequence-nr	The sequence number of this notification from the HMC. This number starts at 0 when the HMC is started, and is incremented each time a notification message is published to any API session.
object-uri	The current value of the **object-uri** property (canonical URI) of the managed object for which the notification is being emitted.
object-id	The current value of the **object-id** property (durable unique identifier) of the managed object.
element-uri	The current value of the **element-uri** property of the element object for which the notification is being emitted. This message property is included only when the message pertains to an element object of a managed object.
element-id	The current value of the **element-id** property (local identifier) of the element object. This message property is included only when the message pertains to an element object of a managed object.
class	The current value of the **class** property (kind of object) of the object, i.e. the kind of object for which the notification is being emitted.
name	The current value of the **name** property (display name) for the object to which the notification pertains. **Note:** In some circumstances the **name** property may be unavailable, in which case this field is set to an empty string. This may occur, for example, if a property change occurs and is to be reported on very shortly before (essentially concurrent with) the removal of that object from the inventory.

When a notification message pertains to an element object, the message includes **element-uri** and **element-id** fields in addition to **object-uri** and **object-id** fields. The element-* fields identify the element object instance while the object-* fields identify the containing managed object instance. In this case, the **class** field provides the class of the element object, and the **name** field provides the name of the element object.

When a notification message pertains to a managed object, the message contains **object-uri** and **object-id** fields but not the element-* fields and the **class** field provides the class of the managed object and the **name** field provides the name of the managed object.

Status change notification

A Status Change notification is emitted by the API to report changes to the **status** property of a managed object.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"status-change"**.

The body of a Status Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
change-reports	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
old-status	String	The old (previous) value of the **status** property for the object. The value of this field will be one of the possible enumeration values for the **status** property as defined for this class of object.
old-additional-status	String	The old (previous) value of the **additional-status** property for the object. The value of this field will be one of the possible enumeration values for the **additional-status** property as defined for this class of object.
new-status	String	The new (current) value of the **status** property for the object. The value of this field will be one of the possible enumeration values for the **status** property as defined for this class of object.
new-additional-status	String	The new (current) value of the **additional-status** property for the object. The value of this field will be one of the possible enumeration values for the **additional-status** property as defined for this class of object.
has-unacceptable-status	Boolean	The value of the **has-unacceptable-status** property of the object, based on its new status. If true, the object is now considered to have unacceptable status because its current status is not one of the configured acceptable status values for this object.

Property change notification

A Property Change notification is emitted by the API to report changes to the properties of a managed object where the data model description indicates that modification notification support (qualifier "pc") is available for that property.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	Contains the value **"property-change"**.
property-names	Blank-separated list of the names of the properties for which change reports are provided in the body of this notification message. The list always includes a leading and trailing blank so that a property name can be specified as a blank-delimited word in a message selector to avoid matching unintended properties that have the desired property name as a substring.

The body of a Property Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
change-reports	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
property-name	String	The name of the property (as specified in the object's data model) that has changed.
old-value	Based on model	If the property is not a container-type or write-only property, this field contains the old (previous) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete previous value. Rather, it provides an array of entries that have been removed from the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been removed, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null.
new-value	Based on model	If the property is not a container-type or write-only property, this field contains the new (current) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete new value. Rather, it provides an array of entries that have been added to the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been added, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null.

Inventory change notification

An Inventory Change notification is emitted by the API to report the addition or removal of a managed object to/from the current inventory of resources that are being managed by zManager. This occurs when managed resources are created or deleted, but also may occur in other situations, such as when zManager reestablishes its inventory of (already-existing) managed resources upon restart of the HMC.

For some kinds of managed objects, an Inventory Change notification is also emitted by the API to report the addition or removal of an element of a managed object. Such notifications do not occur for all elements, but rather only when specifically described in the documentation for a class of managed object.

Because an Inventory Change notification may be generated more than once for the same conceptual object, these notifications cannot be interpreted as designating a resource creation action.

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	Contains the value **"inventory-change"**.
name	Not provided for this notification. Always an empty string.
action	The value **"add"** when the object has been added to the inventory, or **"remove"** when it is being removed.

The body of an inventory change notification is null.

Job completion notification

A Job Completion notification is emitted by the API to report that the processing of an operation that runs asynchronously to the client application has ended.

Asynchronous operations are those that complete with an HTTP status code of 202 (Accepted) when requested by the client. A Job Completion Notification message is sent to the API session that initiated the job when such an operation completes or is canceled, and provides to the client application the URI of the job that has completed or been canceled so the client application can use the Query Job Status operation to obtain results for the job.

Characteristic	Description
Destination	The per-session job notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message properties are provided to allow for message selection:

Message property name	Description
notification-type	The value **"job-completion"**.
job-uri	The URI identifying the asynchronous job that has just completed execution or has been canceled.

The body of a job completion notification is null.

Log entry notification

A Log Entry notification is emitted by the API to report the addition of a log entry to its corresponding console log.

Characteristic	Description
Destination	The audit notification topic or security notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message property is provided to allow for message selection:

Message property name	Description
notification-type	Contains the value **"log-entry"**.

The body of a Log Entry notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
log-entries	Array of objects	An array of nested log-entry-info objects, the format of which is described in Table 443 on page 825. The order in which these objects appear in this array reflects the temporal order in which the log entries were created.

Operating system message notification

An operating system message notification is emitted by the API to report new or refreshed operating system messages.

Characteristic	Description
Destination	One of the os-message-notification topics associated with the API session.

In addition to the common JMS message headers described above, the following additional message property is provided to allow for message selection:

Message property name	Description
notification-type	Contains the value **"os-message"**.

The body of an operating system message notification message is a JSON representation of an object that contains the following field and value:

Field name	Type	Description
os-messages	Array of objects	An array of nested os-message-info objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the messages were created.

Each nested os-message-info object has the following fields and values:

Field name	Type	Description
sequence-number	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message.
message-text	String	The text of the new or refreshed operating system message.
message-id	String	The message identifier of the operating system message.
timestamp	Timestamp	The timestamp represents the date and time when the operating system message was created. A value of -1 is returned if this information is not available from the corresponding operating system.
sound-alarm	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (**true**) or not (**false**).

Field name	Type	Description
is-priority	Boolean	Specifies whether the operating system message is a priority message (**true**) or not (**false**). A priority message indicates a critical condition that requires immediate attention.
is-held	Boolean	Specifies whether the operating system message is a held message (**true**) or not (**false**). A held message is one that requires a response.
prompt-text	String	Specifies the prompt text that is associated with this operating system message or **null** indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.
os-name	String (1-8)	Specifies the name of the operating system that generated this operating system message or **null** indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the partition in which the operating system is running.
is-refresh	Boolean	Specifies whether the message is a new (**false**) or a refresh message (**true**). When the user connects to an os-message-notification topic, operating system messages that already exist are sent as refresh messages, if desired by the user.

Console shutting down notification

A Console Shutting Down notification is emitted by the API to report that the console application is about to be shut down or restarted

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common JMS message headers described above, the following additional message property is provided to allow for message selection:

Message property name	Description
notification-type	Contains the value **"console-shutting-down"**.

The body of a Console Shutting Down notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
console-type	String Enum	Type of console: • **"hmc"** - A Hardware Management Console (HMC). • **"se"** - A Support Element Console (SE).

Field name	Type	Description
invoker	String Enum	The invoker of the shutdown or restart: • **"user"** - Initiated by a user action • **"automation"** - Initiated through an automation interface. • **"firmware"** - Initiated by firmware. • **"other"** - Some other invoker.
reason	String Enum	The reason for the shutdown: • **"firmware-update"** - A firmware update is being performed. • **"problem-recovery"** - Restart is needed to recover from a problem. • **"repair"** - The restart is the result of a repair action. • **"switch"** - The restart was part of a primary/alternate switch operation. • **"other"** - Some other reason.
shutdown-type	String Enum	The type of shutdown being performed: • **"application"** - An application restart only. • **"restart"** - A total console restart (i.e. reboot). • **"shutdown"** - A complete shutdown of the console without a restart. • **"power-off"** - A shutdown of the console followed by a power off of the console hardware.
component	String Enum	The name of the Console component that initiated the shutdown or restart: • **"firmware-management"** - Manages the firmware for the system. • **"problem-analysis"** - Analyzes potential problems encountered for the system. • **"serviceability"** - Performs service actions for the system. • **"pri-alt-support-element"** - Manages the execution of the redundant Support Elements. • **"other"** - Some other component.
delay-tolerance	Integer	The time in seconds that the Console will allow for shutdown delays before the shutdown associated with the notification occurs: • 0 - Delays for this shutdown are not allowed. • > 0 - The maximum time, in seconds, the shutdown can be delayed. • < 0 - Any valid delay time is allowed.
time-until-start	Integer	The time in seconds until the associated shutdown or restart operation will be started: • 0 - The operation will be started immediately. • > 0 - The amount of time an application has before the operation starts. • < 0 - The operation will be started when there are no remaining applications delaying the shutdown.

Disabled wait notification

A Disabled Wait notification is emitted by the API to report that a Logical Partition object has entered a disabled wait state. [Added by feature **bcpii-notifications**]

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common message characteristics and properties the following property is provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"disabled-wait"**.

The body of a Disabled Wait notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
cpc-object-uri	String/ URI	The **object-uri** for the CPC object hosting the logical partition.
cpc-object-id	String (36)	The unique **object-id** for the CPC object hosting the logical partition.
program-status-word	String	The disabled wait program status word (PSW).
partition-id-num	Integer	The partition identifier.
scp-initiated-reset	Boolean	A value of true indicates the disabled wait was due to an SCP initiated reset.
processor-num	Integer	The number of the processor encountering the disabled wait.

Capacity change notification

A Capacity Change notification is emitted by the API to report changes in the processing capacity for a CPC object. [Added by feature **bcpii-notifications**]

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common message characteristics and properties the following property is provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"capacity-change"**.

The body of a Capacity Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
capacity-change-type	String Enum	The type of capacity change. • **"fenced-book"** - The change was due to a processor book being fenced. • **"defective-processor"** - The change was due to a defective processor. • **"concurrent-book-replace"** - The change was due to a concurrent processor book replace. • **"concurrent-book-add"** - The change was due to a concurrent processor book add. • **"check-stop"** - The change was due to a system check stop. • **"changes-allowed"** - The change was due to automated capacity changes being enabled. • **"changes-not-allowed"** - The change was due to automated capacity changes being disabled.

Capacity record change notification

A Capacity Record Change notification is emitted by the API to report changes related to a capacity record object. [Added by feature **bcpii-notifications**]

Characteristic	Description
Destination	The per-session object notification topic for each API session that is authorized to receive the notification.

In addition to the common message characteristics and properties the following property is provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"capacity-record-change"**.

The body of a Capacity Record Change notification message is a JSON representation of an object that contains the following fields and values:

Field name	Type	Description
capacity-record-change-type	String Enum	The type of capacity change. • **"add"** - A capacity record was added. • **"delta"** - A capacity record was modified. • **"level"** - A capacity record has changed its level of activation. • **"priority-pending"** - Additional capacity has been added for the capacity record with priority, but not enough resources were available to allow for all the capacity specified to be put into effect. As resources become available, they will be added for this record in order to completely satisfy the original request for additional capacity. • **"other"** - Some other unexpected type of capacity record change. • **"delete"** - A capacity record was deleted. • **"accounting"** - An accounting change has occurred.

Server-Sent Events (SSE)

Server-Sent Events is an HTTP specification for subscribing to a data stream and receiving event messages asynchronously in return. Implementations for both clients and server handlers are included with Java EE's JAX-RS package. Open-Source client implementations can also be found in various other languages.

On the HMC, clients subscribe with a two-step process. First, configure an event stream for what event types and filters are desired. Finally, open the configured stream. Both steps are HTTP Web Services API operations described in "Create Server-Sent Events Stream" on page 134.

A single client session can create and open multiple streams and can also update an existing stream's configuration on the fly. This could be useful for updating property change filters in response to an inventory add event, for example.

Every event has an identifier property, as defined by SSE specification. For HMC events, this is a sequence number, unique per stream and starting at 0 when a stream is created. It is incremented for every event message sent. SSE uses this to attempt to retry event notifications in the event of a failure. However, this behavior is not guaranteed, and therefore clients are also able to use this to determine they may be out of sync if they observe a sequence number skip.

The HMC may emit comment messages to keep a stream alive if an event has not been sent for a certain amount of time. These messages will only contain an ignorable comment; the id, name, and data will all be null.

Security considerations

Creating an SSE stream requires a fully authenticated API session. The HMC will therefore ensure only events the user is authorized to receive are sent to that stream (objects, log entries, etc.).

However, in general SSE clients do not support setting request header values when opening a stream, as is the common method of authorizing an API operation. Therefore, the stream identifier is used as the authorization token for opening the stream. Because it is sent as a query parameter over an encrypted connection, it will not be sent in the clear. Additionally, it is tied to the session and will therefore expire when the session expires, either through a `Logoff` operation or the session timing out.

Initial events

A client can optionally receive an initial set of events describing the current state of all configured registrations when opening a stream. This allows a client to get the current snapshot in the same stream and listener implementation rather than needing to both subscribe for changes and perform various Get Properties and/or Get Inventory operations.

All initial events are guaranteed to be sent prior to any asynchronous event notifications. Any change events that occur during initial event processing are held by the HMC and then sent once that is complete, in temporal order, to ensure no updates are missed. Additionally, special bookend events are sent to note the beginning and end of sending initial events. This can be used by clients to know when they have received a complete snapshot of the current state. Such events will be sent with the Console object as its source, and an event name of "initial-events-start" or "initial-events-end".

When a stream is opened with initial events requested, that is remembered until the stream is closed. There are additional circumstances when initial events could be sent after the first set are sent upon opening. In these cases, once again the HMC will pause any asynchronous notifications to gather the current snapshot of just the new event data to send first. Such situations are as follows:

- If the stream configuration is changed while open, then any added registrations will cause an initial set of events to be sent for them. The HMC will compare the old and new configurations to determine anything new, such as a new object for Property Change events.
- If a stream is configured to listen for Status Change, Property Change, or OS Message events on all objects or a class of objects, and such an object or element is added to inventory, then initial events will

be sent for the object or element. In this case, if the stream is also registered for inventory events, the HMC guarantees the inventory add event will be sent prior to the new initial events.

To fully support initial events retrieving all that a client may want, stream configurations can contain properties that do not support Property Change events, i.e., those not marked with a (pc) qualifier. Of course, change notifications would never be sent for such properties, but they would be included in the initial events notification.

Notification event formats

Several types of notification events are provided by the API. The SSE events created for all types of notifications share a common set of properties and fields, which are extended with additional data fields that vary by the type of notification.

Event properties

The following properties are standard for SSE messages. The table describes how they will be used by the HMC.

Table 45. SSE notification event properties

Event property	Description
`id`	The sequence number of this event message, unique per stream. It starts at 0 and is incremented for every event sent. This is null for keep-alive comment messages.
`name`	The name of this event message. It is the type of event, and either correlates to the configurable **event-names** String Enum values in the `Create Server-Sent Events Stream` operation, or the special events of "initial-events-start" and "initial-events-end". This is null for keep-alive comment messages.
`data`	The JSON object data payload describing the event. The content varies by the type of notification, as described below. This is null for keep-alive comment messages.
`comment`	An ignorable String for keep-alive comment messages. This is null for regular event messages.

Common event data fields

All regular SSE messages contain a JSON object as the data property. The following are the JSON object fields supported by all event types:

Table 46. SSE common event properties

Field name	Type	Description
`stream-id`	String	The unique identifier of the stream to which this event is being sent, as returned when the stream was configured and was passed to the `Open Server-Sent Events Stream` operation.
`event-sequence-nr`	Integer	The sequence number of this event message. It is the same as the event's **id** property, included in the JSON object for convenience.

Table 46. SSE common event properties (continued)		
Field name	**Type**	**Description**
`event-name`	String	The name of this event message (type of event). It is the same as the event's **name** property, included in the JSON object for convenience.
`is-initial`	Boolean	True if this event is part of the requested initial events, or false if it is an asynchronous notification of a change.
`object-uri`	String/URI	The current value of the **object-uri** property (canonical URI) of the managed object for which the notification is being emitted.
`object-id`	String	The current value of the **object-id** property (durable unique identifier) of the managed object.
`element-uri`	String/URI	The current value of the **element-uri** property of the element object for which the notification is being emitted. This message property is included only when the message pertains to an element object of a managed object.
`element-id`	String	The current value of the **element-id** property (local identifier) of the element object. This message property is included only when the message pertains to an element object of a managed object.
`class`	String	The current value of the **class** property (kind of object) of the object, i.e. the kind of object for which the notification is being emitted.
`name`	String	The current value of the **name** property (display name) for the object to which the notification pertains. **Note:** In some circumstances the **name** property may be unavailable, in which case this field is set to an empty string. This may occur, for example, if a property change occurs and is to be reported on very shortly before (essentially concurrent with) the removal of that object from the inventory.

When a notification message pertains to an element object, the message includes **element-uri** and **element-id** fields in addition to **object-uri** and **object-id** fields. The element-* fields identify the element object instance while the object-* fields identify the containing managed object instance. In this case, the **class** field provides the class of the element object, and the **name** field provides the name of the element object.

When a notification message pertains to a managed object, the message contains **object-uri** and **object-id** fields but not the element-* fields and the **class** field provides the class of the managed object and the **name** field provides the name of the managed object.

Initial events start/end notification

An Initial Events Start/End notification is emitted by the API to report that the forthcoming set of events are initial events, or the sending of the set of initial events is complete, respectively. The event name is either "initial-events-start" or "initial-events-end", and the event source is the Console object.

Note that receiving these special bookend events does not guarantee receiving any actual initial events. For example, a client may register for properties of an object that is not currently available, in which case no initial events would be sent. Such a client would be able to use the bookend events to determine that.

No additional fields are provided beyond the common fields.

Status change notification

A Status Change notification is emitted by the API to report changes to the **status** property of a managed object.

In addition to the common fields, the following are also included:

Field name	Type	Description
`change-reports`	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
`old-status`	String	The old (previous) value of the **status** property for the object. The value of this field will be one of the possible enumeration values for the **status** property as defined for this class of object. This will be null for initial events.
`old-additional-status`	String	The old (previous) value of the **additional-status** property for the object. The value of this field will be one of the possible enumeration values for the **additional-status** property as defined for this class of object. This will be null for initial events.
`new-status`	String	The new (current) value of the **status** property for the object. The value of this field will be one of the possible enumeration values for the **status** property as defined for this class of object.
`new-additional-status`	String	The new (current) value of the **additional-status** property for the object. The value of this field will be one of the possible enumeration values for the **additional-status** property as defined for this class of object.
`has-unacceptable-status`	Boolean	The value of the **has-unacceptable-status** property of the object, based on its new status. If true, the object is now considered to have unacceptable status because its current status is not one of the configured acceptable status values for this object.

Property change notification

A Property Change notification is emitted by the API to report changes to the properties of a managed object where the data model description indicates that modification notification support (qualifier "pc") is available for that property.

In addition to the common fields, the following are also included:

Field name	Type	Description
`property-names`	Array of string	An array of the names of the properties for which change reports are included in this event object.
`change-reports`	Array of objects	An array of nested change-report objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the changes occurred.

Each nested change-report object has the following fields and values:

Field name	Type	Description
`property-name`	String	The name of the property (as specified in the object's data model) that has changed.

Field name	Type	Description
`old-value`	Based on model	If the property is not a container-type or write-only property, this field contains the old (previous) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete previous value. Rather, it provides an array of entries that have been removed from the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been removed, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null. This will be null for initial events.
`new-value`	Based on model	If the property is not a container-type or write-only property, this field contains the new (current) value of the property for the object. The value of this field will be of the data type indicated for this property in the object's data model. If the property is a container-type property (i.e. marked with the (c) qualifier), this field does not provide the complete new value. Rather, it provides an array of entries that have been added to the value of the container property. The value of these entries will be of the data type indicated for the property in the object's data model. If no entries have been added, null is provided. If the property is a write-only property (i.e. marked with the (wo) qualifier), this field does not provide the value of the property. Rather, this field always contains null.

Inventory change notification

An Inventory Change notification is emitted by the API to report the addition or removal of a managed object to/from the current inventory of resources that are being managed by the HMC. This occurs when managed resources are created or deleted, but also may occur in other situations, such as when the HMC reestablishes its inventory of (already-existing) managed resources upon a restart.

For some kinds of managed objects, an Inventory Change notification is also emitted by the API to report the addition or removal of an element of a managed object. Such notifications do not occur for all elements, but rather only when specifically described in the documentation for a class of managed object.

Because an Inventory Change notification may be generated more than once for the same conceptual object, these notifications cannot be interpreted as designating a resource creation action.

In addition to the common fields, the following are also included:

Field name	Type	Description
`action`	String Enum	The value **"add"** when the object has been added to the inventory or for initial events, or **"remove"** when it is being removed.

Job completion notification

A Job Completion notification is emitted by the API to report that the processing of an operation that runs asynchronously to the client application has ended.

Asynchronous operations are those that complete with an HTTP status code of 202 (Accepted) when requested by the client. A Job Completion notification message is sent to the API session that initiated the job when such an operation completes or is canceled and provides to the client application the URI of the job that has completed or been canceled so the client application can use the Query Job Status operation to obtain results for the job.

Note that Job Completion notifications are not supported in initial events.

In addition to the common fields, the following are also included:

Field name	Type	Description
`job-uri`	String/ URI	The URI identifying the asynchronous job that has just completed execution or has been canceled.

Log entry notification

A Log Entry notification is emitted by the API to report the addition of a log entry to its corresponding console log. The format of the event object is the same for entries of the HMC's audit, security, and event logs.

In addition to the common fields, the following are also included:

Field name	Type	Description
`log-entries`	Array of objects	An array of nested log-entry-info objects, the format of which is described in Table 443 on page 825. The order in which these objects appear in this array reflects the temporal order in which the log entries were created.

Operating system message notification

An operating system message notification is emitted by the API to report new or refreshed operating system messages.

In addition to the common fields, the following are also included:

Field name	Type	Description
`os-messages`	Array of objects	An array of nested os-message-info objects, the format of which is described in the next table. The order in which these objects appear in this array reflects the temporal order in which the messages were created.

Each nested os-message-info object has the following fields and values:

Field name	Type	Description
`sequence-number`	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message.
`message-text`	String	The text of the new or refreshed operating system message.
`message-id`	Long	The message identifier of the operating system message.
`timestamp`	Timestamp	The timestamp represents the date and time when the operating system message was created. A value of -1 is returned if this information is not available from the corresponding operating system.
`sound-alarm`	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (**true**) or not (**false**).

Field name	Type	Description
`is-priority`	Boolean	Specifies whether the operating system message is a priority message (**true**) or not (**false**). A priority message indicates a critical condition that requires immediate attention.
`is-held`	Boolean	Specifies whether the operating system message is a held message (**true**) or not (**false**). A held message is one that requires a response.
`prompt-text`	String	Specifies the prompt text that is associated with this operating system message or **null** indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.
`os-name`	String (1-8)	Specifies the name of the operating system that generated this operating system message or **null** indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the partition in which the operating system is running.

Chapter 5. Data model definitions

This chapter covers data model concepts and shared data model schema elements.

Data model concepts

zManager provides resource management and control functions for the various resources known to the HMC. In performing these functions, zManager establishes a separation between those aspects of resource management that are handled entirely by system firmware, and the other aspects for which customer or installation visibility, configuration and control is appropriate.

In order to specify the external aspects in a succinct way, the Web Services API is described in this document in terms of a conceptual data model that it offers for the resources that it manages. This data model is an information structuring technique that conceptually defines the kinds of resources that are managed by zManager and for each, the information that is available for and the operations that can be performed on resources of that kind. This data model is intended to provide the complete perspective that clients of the API can have regarding the logical resources of the system while insulating them from implementation details.

Objects in the data model

The manageable resources of the environment are represented in the management system as entities referred to as objects. Each distinct manageable resource is represented by a separate object instance, and the life cycle of an instance corresponds with the lifecycle of the manageable resource it represents. For example, for physical entities, such as a CPC, the object that represents it is created implicitly when the physical entity is attached to and configured to be part of the system. This object continues to exist so long as the CPC is managed by the HMC.

There are different kinds of manageable physical or logical resources in the system, and each kind manifests different observable characteristics. As a result, there are different classes of objects in the data model. Objects of the same class represent the same kind of resource and provide a defined set of *properties* that capture the attributes of that kind of resource that the Web Services API exposes.

Managed objects and element objects

The object classes defined in the data model fall into one of two main categories: managed object classes (or simply managed objects), and element object classes (element objects).

These two categories are very similar in that they are both, ultimately, unordered collections of named properties that capture the key attributes of a resource instance. The categories differ primarily in how prominently they are handled in the API: the way that instances of them are designated to perform operations on them, and the degree to which API facilities such as inventory and change notification can be offered for objects in that category.

Managed objects are the first-class entities in the data model and the API. They represent the primary manageable resources of the system, such as CPCs, logical partitions, adapters and partitions. These kinds of objects typically appear prominently in the main displays of the HMC user interface.

Instances of managed objects are registered and indexed in the zManager managed object registry, and thus can be directly referenced by URIs that form a relatively "flat" namespace. The URI of a managed object designates its object instance based on its class and a unique, durable, UUID-based identifier called an object ID. For example, the URI of a CPC is of the form `/api/cpcs/{cpc-object-id}` where the identifier at the end of the URI is globally unique. Inventory change, property change, and status change notifications can be generated for managed objects.

In comparison, element objects represent the secondary or more-detailed aspects of the system. Examples include the Load Activation Profiles of a CPC, or the NICs of a Partition. These kinds of entities

© Copyright IBM Corp. 2022, 2024

do not generally appear in the main displays of the HMC user interface, but rather are displayed only within particular management tasks offered by the UI.

Instances of element objects are not directly registered in the zManager object registry, but rather are associated with or “attached to” some containing or related managed object instance. As a result, access to these elements is indirect, through the containing managed object. The URIs that designate element objects are hierarchical in nature, with the leftmost part of the URI identifying the managed object to which the element is attached. For example, the URI for a HBA of a Partition is of the form `/api/partitions/{partition-object-id}/hbas/{hba-id}` in which the *{hba-id}* at the end is only necessarily unique within the context of the related partition. Inventory, property and status change notifications are not always directly provided for element objects. In such cases, when changes to elements are reported, those changes are done through property change notifications emitted for the associated managed object.

Properties in the data model

Objects in the management system contain, fundamentally, unordered collections of name/value pairs called properties that capture the key characteristics of the manageable resources they represent. The defined set of named properties that are maintained for a particular kind of resource constitutes the specification of the data model class for that kind of resource.

As a result, in the chapters that follow, the description of the management interfaces for a class of resource begins with a data model section that specifies the properties that are exposed by the API for that kind of resource.

Each property has a name, a data type, and a semantic description in prose.

The property name is the programmatic identifier of the property. This identifier is used within requests and responses to indicate that a field represents a particular property of the data model. It is the “name” part of the name/value pair that is the property.

The property data type indicates the kind of information that can be represented by the property, just as a variable's data type indicates the kind of information that can be stored in a variable. The data type provides information on the nature of the "value" part of the name/value pair that is the property.

Property characteristics

Properties are classified as being either writable or read-only from the perspective of an API client application.

Writable properties are ones that can have their values read by Get <class> Properties or similar operations and can also have their values directly changed by Update <class> Properties operations. Properties that are classified as write-only can have their values directly changed by Update <class> Properties operations, but cannot have their values read by using Get <class> Properties or similar operations. Properties that are classified as read-only can have their values read by using Get <class> Properties or similar operations, but cannot have their values changed directly.

Although properties that are classified as read-only cannot have their values changed directly, their values may nonetheless be affected by other operations supported by a class of object. For example, a class of object might include an **is-enabled** property that is classified as read-only because the enabled state of the resource cannot be affected by a simple Update <class> Properties operation on that **is-enabled** property. However, this object might also define a Change State or Enable operation that can perform this enabling, and as a side effect will alter the value of the **is-enabled** property.

In addition to the read-only vs. writable classification, properties defined for a managed object also can differ in whether changes to them result in property change or status change notifications being emitted for the managed object or not. For properties that have property or status change support, these notifications are emitted asynchronously by the API any time the value of the property changes, whether that change was made through this API, the HMC UI, or implicitly by the system. Changes to the values of properties for which change notification support is not provided do not result in such notifications.

Most objects have properties that are primarily configuration data. However, for some objects, certain of those configuration-related properties may at times also have a transient, runtime counterpart property in effect whose value can be different than the preserved configuration or base value. That transient, runtime counterpart property is known as an "effective" property and is identified as such in the object's data model by the (e) qualifier. The name of an effective property is formed by prepending "effective-" to the name of its corresponding base property. Whether an object's effective properties are applicable at any point in time is determined by the state of the object or the state of the system or related objects and is described in that object class' section of this document. The object's **effective-properties-apply** property indicates whether effective properties are currently applicable. If **effective-properties-apply** is **false**, the value of an effective property is the same as its corresponding base property and the effective property may not be altered by an Update <class> Properties operation.

In some cases, an object in the management system may have properties whose values are unwieldy to provide or expensive to obtain and further may not be of general interest in typical API client use cases. To allow the handling of such properties to be optimized they are represented by a special kind of property termed a pseudo property. A pseudo property is conceptually one of the characteristics of a class of object and is thus documented in the data model for the object (with a (p) qualifier). However, the name and value of a pseudo property is not included in the response to a Get <class> Properties operation or in the inventory service data for that class of object. Instead, class-specific operations are provided in order to obtain the current value of the property when needed by an API client. As for normal properties, property change notifications may be generated for changes to a pseudo property if indicated in the data model.

There are times when it is convenient to provide a property of a related object as a property of itself. Such properties are referred to as ancillary properties (with an (a) qualifier). For example, an object that is hierarchically a child of another object may include its parent’s name as a property of its own. This property is not really part of the data model of the child's class of objects but may be provided as such for convenience. In these cases, Property Change support is not provided on the ancillary property, even if it is supported by the class of object to which it belongs.

In the tables of properties that appear within this document, the characteristics of properties are indicated by qualifier annotations in parenthesis following the property name. The qualifiers have the following meanings:

Qualifier notation	**Description**
(w)	The property is a writable property. Any property that lacks this qualifier and the (wo) qualifier is considered read-only and thus is not directly modifiable.
(wo)	The property is a write-only property. Any property that lacks this qualifier and the (w) qualifier is considered read-only and thus is not directly modifiable.
(ro)	Although this property is writable when present in other managed object classes, it is read only in this class. This qualifier is only used when a managed object class specializes the definition of a base managed object property and overrides the writable characteristic of the base definition.
(p)	The property is a pseudo property. Its current value is omitted from the response to a Get <class> Properties operation or inventory service data for the object, but can be obtained by a class-specific operation.
(pc)	Change to this property's value will result in Property Change notifications.
(c)	The property is a container-type property for which deltas (changes) are reported in Property Change notifications rather than complete old and new values.
(sc)	Change to this property will result in Status Change notifications.

Qualifier notation	Description
(mg)	This property represents a performance or utilization metric of the object that is included in a metric group available through the Metrics Service of this API. The value of this property may change very frequently and, therefore, property change notifications are not emitted for changes to this property. Client applications interested in obtaining metric information frequently should obtain this information through use of the Metrics Service of this API.
(e)	The property is an effective property.
(a)	The property is an ancillary property of the class of objects in which it is defined and is owned by a related class of objects.

Shared data model schema elements

The data-model schema fragments in this section define groups of properties that are used in common ways in specifying the data models for the managed object classes defined in the API.

The description of the data model for a specific object class specifies the shared schema elements it is incorporating within the data model section of that description, if any. It will also include a description of the specializations that apply to that class's use of the shared schema, such as additional constraints on properties, class-specific values for properties, etc.

Base managed object properties schema

This data-model fragment contains the basic properties that are present in the representation of many of the managed object types that represent manageable resources.

Name	Qualifier	Type	Description
object-uri	—	String (1-255)	The canonical URI path that designates this managed object instance and serves as the primary reference and retrieval key for this instance. The URI path is formed based on a unique and permanently-assigned object ID (see the **object-id** property in the next row of this table), and as result, an object's URI path will not change as a result of changes to properties of the object. Further, this canonical URI path is independent of the containment hierarchy and thus will not change if this object instance is moved within the hierarchy.
object-id	—	String (36)	The object identifier for the managed object instance. This value is unique in space and time, and is permanently associated with this instance while it is managed by this HMC. (If the instance is removed from this HMC and later managed by another HMC, it will have a new and different object identifier when managed by that other HMC.) It is generated by zManager and assigned when the managed resource is created or first discovered, and is immutable thereafter. As example, a managed object's object ID will not change as a result of changes to display name, changes in the location of this resource in the containment hierarchy, or across restarts of the HMC.
parent	—	String (1-255)	The canonical URI path of the managed object that is conceptually the parent of this object in the containment hierarchy. This property is null for objects that do not have a parent.

Name	Qualifier	Type	Description
class	—	String (1-32)	The class of resource represented by this managed object. Each distinct class of resource has a different type name, while all instances of the same type share the same type name. The specific value used for a class of object is specified in the data model section for that object type. Example: **"logical-partition"**.
name	(w)(pc)	String (1-64)	The current display name of the managed object as defaulted or specified for the object. This is the simple name of this object, not qualified by containment hierarchy. This name must consist only of alphanumeric characters and the following special characters: period (.), hyphen (-), at sign (@), underscore (_), and space. It must not begin or end with a space. Some resource types do not support the setting of a user-assigned display name. For such objects, this property is not settable, and instead always provides a name assigned by the HMC or SE.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about this managed resource. This information is retained for the resource and may be shown as part of the object's details on the user interface, but is otherwise not generally used by zManager. This property may be null.
is-locked	(pc)	Boolean	The object is locked and thus disruptive actions or tasks cannot be performed on it.
effective-properties-apply	—	Boolean	The object is currently in a state in which effective properties are applicable. As this property is only meaningful for object classes whose data model includes effective properties, it is only included for those object classes.

Operational status properties

Many (but not all) classes of managed objects support the concept of operational status. That is they maintain information about the current functional state (Not Communicating, Not Operating, etc.) of the managed resource and whether that current functional state is considered acceptable (not alert causing) or not. If a class of object supports the operational status concept, it provides the standard properties defined in the following table (referred to as the operational status properties) in addition to those defined earlier in this section.

Unless stated to the contrary, any object class data model that includes the base managed object properties schema should be understood to also provide these operational status properties as well. For object classes for which that is not the case, the data model description will specifically point out that operational status and thus these operational-status-related properties are not provided for that object.

The operational status properties are as follows:

Name	Qualifier	Type	Description
status	(sc)	String Enum	The current operational status of the managed resource. The possible status values vary by managed object class and are specified in the description of each managed object class that provides this property.
additional-status	(sc)	String Enum	A qualifier to the **status** property used by selected object classes to provide finer grained operational status tracking.
acceptable-status	(w)(pc)	Array of String Enum	The set of operational status values that the managed resource can be in and be considered to be in an acceptable (not alert causing) state.

Name	Qualifier	Type	Description
has-unacceptable-status	(sc)	Boolean	If true, the current operational status of the managed resource is not one of the acceptable statuses for this resource.

Chapter 6. Features

This chapter describes API features and firmware features.

API features

Beginning with API version 4.10, logically related API changes and additions are grouped into "API features" and assigned a name. Rather than use the API version and SE version and MCL level to determine if certain API functionality is available, API clients must use the `List Console API Features` and `List CPC API Features` operations. Those operations return a list of the API features that are available on the HMC and SE, respectively. Some feature implementations may be entirely contained on the HMC or on the SE, while others involve both. Because of that, some feature names will only be returned for the Console or the CPC while others will be returned for both. API features whose implementation involves both will not be fully available to API clients unless both the HMC and the relevant CPC report that the feature is available.

An API feature may include new or changed:

- properties in object data models
- operations
- notification messages
- metric groups
- inventory categories and classes
- other aspects of the API visible to API users

The following are the feature names that may be returned in the response body of the `List Console API Features` and `List CPC API Features` operations.

Table 47. API Features

Name	Description	Console	CPC
adapter-network-information	List Permitted Adapters on the SE with network-port additional-properties	✓	✓
bcpii-notifications	Support for receiving Asynchronous Notifications via BCPii v2		✓
cpc-delete-retrieved-internal-code	Delete retrieved CPC MCLs that are not installed. Perform retrieval from FTP server in CPC Single Step Install.	✓	✓
cpc-install-and-activate	Support to install and activate CPC MCLs	✓	✓
create-delete-activation-profiles	Support for creating and deleting activation profiles	✓	✓
dpm-ctc-partition-link-management	DPM Support of FICON Channel to Channel interconnect technology (CTC) Partition Links	✓	✓
dpm-hipersockets-partition-link-management	DPM Support of Hipersockets Partition Links	✓	✓
dpm-smcd-partition-link-management	DPM Support of SMC-D Partition Links	✓	✓

© Copyright IBM Corp. 2022, 2024

Table 47. API Features (continued)			
environmental-metrics	Query system and partition environmental metrics	✓	✓
hmc-delete-retrieved-internal-code	Delete retrieved HMC MCLs that are not installed. Perform retrieval from FTP server in Console Single Step Install.	✓	
ldap-direct-authentication	Support for *direct* authentication when accessing user and group membership information on an LDAP server	✓	
mobile-enhanced-push	Allow enabling enhanced push notifications for HMC Mobile in HMC Mobile Settings	✓	✓
oem-hmc-ids	Numeric IDs for OEM HMCs	✓	
pmg-child-management-permission	Allow child management permission to Pattern Match Groups	✓	
rc-409-15	New common HTTP status 409 with reason code 15	✓	✓
rcl-history	Support for tracking of Remote Code Load history	✓	✓
rcl-progress	Support for determinate progress of Remote Code Loads	✓	✓
remote-firmware-update-rc404-4	HTTP status 404 with reason code 4 for operations handling remote firmware updates on a CPC.	✓	
report-a-problem	Support for reporting Console and CPC problems	✓	✓
secure-boot-with-certificates	Support for z/OS and Linux validated boot including secure boot capability available for IPL from ECKD DASD with user-provided signature validation keys	✓	✓
secure-execution-key-management	Support for importing Secure Execution key bundles and deleting secondary keys	✓	✓
switch-support-elements	Support for switching the Primary Support Element and the Alternate Support Element	✓	✓

Each place in this document changed by a feature will indicate which API feature made the change. Changes or additions made by a given feature are indicated with the following text:

- For additions: [Added by feature ***feature-name***]
- For changes: [Updated by feature ***feature-name***]

Firmware features

Starting with HMC version 2.14.0 and API version 2.23, features can be enabled for specific objects. To indicate this, an available-features-list property is introduced to the objects that are affected. These features may be enabled by default from a specific HMC or SE version onwards or enabled by using standard feature enablement mechanisms.

These features affect certain API operations as documented here. API clients should query the enablement of these features on a system and choose the operations accordingly.

The enablement of a feature might mean that the API clients have to use new API operations or new properties in existing API operations. It might also indicate that some of the existing API operations and properties will not be supported.

If an API operation is not supported when a feature is enabled, invoking the API operation on an object where the feature is enabled would result in a standard status code 409 (Conflict) with a standard reason code 12.

If an API operation is supported only when a feature is enabled, invoking the API operation on an object where the feature is disabled would result in a standard status code 409 (Conflict) with a standard reason code 13.

If an API operation targeting an object whose existence is controlled by a feature is invalid given the enablement of the feature, a standard status code 404 (Not Found) with standard reason code 1 (for managed objects) or 5 (for element objects) is returned.

The following section describes the features that are currently available.

dpm-storage-management

This feature is applicable for the CPC and Partition objects. The `Get CPC Properties` and `Get Partition Properties` operations can be used to query if the feature is enabled or disabled.

When this feature is enabled, management of FICON storage is available. FCP and FICON virtual storage resources are defined in Storage Groups which are then attached to Partitions. A Partition that has this feature enabled has no HBAs visible to an API client or on the UI.

When this feature is disabled, FICON storage is not available and FCP virtual storage resources (HBAs) are attached directly to Partitions.

The following API operations are affected and will return status code 409 (Conflict) with reason code 12 when they are invoked on an object on which the **"dpm-storage-management"** feature is enabled.

- `Create HBA`
- `Export WWPN List`
- `Dump Partition`

The following API operations are affected and will return status code 404 (Not Found) with reason code 5 when they are invoked on an HBA object on which the **"dpm-storage-management"** feature is enabled.

- `Delete HBA`
- `Update HBA Properties`
- `Get HBA Properties`
- `Reassign Storage Adapter Port`

There are new API operations introduced with the feature. They are defined in the following sections under Chapter 10, “Dynamic Partition Manager (DPM),” on page 195:

- “Storage Site operations summary” on page 202
- “Storage Fabric operations summary” on page 202
- “Storage Switch operations summary” on page 203
- “Storage Subsystem operations summary” on page 203
- “Storage Control Unit operations summary” on page 204
- “Storage Group operations summary” on page 205

In addition, the following new API operations are introduced with the feature and are defined in “Partition operations summary” on page 197:

- `Start Dump Program`
- `Attach Storage Group to Partition`

- `Detach Storage Group from Partition`

These operations will return status code 409 (Conflict) with reason code 13 when they are invoked on an object on which the **"dpm-storage-management"** feature is disabled.

dpm-fcp-tape-management

This feature is applicable for the CPC and Partition objects. The `Get CPC Properties` and `Get Partition Properties` operations can be used to query if the feature is enabled or disabled.

When this feature is enabled, management of FCP tape storage is available. FCP virtual tape resources are defined in tape links which are then attached to partitions. When this feature is disabled, the Tape Link and Tape Library objects are not available.

There are API operations introduced with the feature. They are defined in "Tape Library operations summary" on page 207 and "Tape Link operations summary" on page 207 and include:

- Tape Link object operations:
  - `List Tape Libraries`
  - `Undefine Tape Library`
  - `Get Tape Library Properties`
  - `Update Tape Library Properties`
  - `Request Tape Library Zoning`
  - `Discover Tape Libraries`
- Tape Library object operations:
  - `List Tape Links`
  - `Create Tape Link`
  - `Get Tape Link Properties`
  - `Modify Tape Link Properties`
  - `Delete Tape Link`
  - `Add Adapter Ports`
  - `Remove Adapter Ports`
  - `Replace Adapter Port`
  - `Resend Request`
  - `List Virtual Tape Resources of a Tape Link`
  - `Get Virtual Tape Resource Properties`
  - `Update Virtual Tape Resource Properties`
  - `Get Partitions for a Tape Link`
  - `Get Tape Link Histories`
  - `Update Tape Link Environment`
  - `Get Tape Link Environment Report`

In addition, the following API operations are introduced with the feature and are defined in "Partition operations summary" on page 197:

- `Attach Tape Link to Partition`
- `Detach Tape Link from Partition`

These operations will return status code 409 (Conflict) with reason code 13 when they are invoked on an object on which the **"dpm-fcp-tape-management"** feature is disabled.

dpm-smcd-partition-link-management

This feature is applicable for the CPC and Partition objects. If the feature is enabled for a CPC, it is enabled for all Partitions of that CPC. The `Get CPC Properties` and `Get Partition Properties` operations can be used to query if the feature is enabled or disabled.

When this feature is enabled, management of SMC-D Partition Links is available. SMC-D devices are defined in Partition Links which are attached to Partitions. When this feature is disabled, Partition Link objects of **type "smc-d"** are not available.

There are API operation introduced with this feature. They are defined in "Partition Link operations summary" on page 208 and include:

- `Create Partition Link`
- `List Partition Links`
- `Delete Partition Link`
- `Get Partition Link Properties`
- `Modify Partition Link`

These operations will return status code 409 (Conflict) with reason code 13 when they are invoked on an object on which the **dpm-smcd-partition-link-management** feature is disabled.

Part 2. General services

Topics in this part describe the general services available for the Web Services API.

Topics covered in this part are:

- Chapter 7, “General API services,” on page 111
- Chapter 8, “Inventory and metrics services,” on page 159
- Chapter 9, “Metric groups,” on page 177

© Copyright IBM Corp. 2022, 2024

Chapter 7. General API services

This chapter describes the services that are provided by the Web Services API for creating and deleting API sessions and performing other general functions.

General API services operations summary

Table 48. General API services: operations summary

Operation name	HTTP method and URI path
“Query API Version” on page 113	`GET /api/version`
“Logon” on page 115	`POST /api/sessions`
“Establish Shared Secret Key” on page 121	`POST /api/sessions/operations/establish-shared-secret-key`
“Provide Requested MFA Information” on page 123	`POST /api/sessions/operations/provide-more-mfa-information`
“Change Logon Password” on page 126	`POST /api/sessions/operations/change-logon-password`
“Verify Logon Password” on page 129	`POST /api/sessions/operations/verify-logon-password`
“Logoff” on page 130	`DELETE /api/sessions/this-session`
“Get Notification Topics” on page 131	`GET /api/sessions/operations/get-notification-topics`
“Create Server-Sent Events Stream” on page 134	`POST /api/sessions/operations/create-server-sent-events-stream`
“Update Server-Sent Events Stream” on page 139	`POST /api/sessions/operations/update-server-sent-events-stream`
“Delete Server-Sent Events Stream” on page 140	`POST /api/sessions/operations/delete-server-sent-events-stream`
“Open Server-Sent Events Stream” on page 141	`GET /api/sessions/operations/open-server-sent-events-stream`
“Get Server-Sent Events Stream Last Event ID” on page 143	`POST /api/sessions/operations/get-server-sent-events-stream-last-event-id` [Added by feature **mobile-enhanced-push**]
“Submit Requests” on page 144	`POST /api/services/aggregation/submit`

© Copyright IBM Corp. 2022, 2024

Table 48. General API services: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Query Job Status" on page 151	`GET /api/jobs/{job-id}`
"Delete Completed Job Status" on page 154	`DELETE /api/jobs/{job-id}`
"Cancel Job" on page 156	`POST /api/jobs/{job-id}/operations/cancel`

Table 49. General API services: URI variables	
Variable	**Description**
{job-id}	The identifier of an asynchronous job associated with this user, as returned in the response of the operation that initiated the job.

Session management services

Almost all operations of the Web Services API are requested and carried out in the context of an API session that is used for determining the client's authority to access managed resources and perform requested operations. It is also used to scope the delivery of asynchronous notifications and manage WebSocket instances. An API session is an HMC concept that is independent of any layers on top of network-related considerations such as a TCP/IP socket connection. As a result, a single API session may span multiple TCP/IP socket connect/disconnect sequences from the same client.

Sessions are created upon request from a client by using the `Logon` operation, and may be explicitly terminated by a client using the `Logoff` operation. Sessions may also be terminated by the HMC due to inactivity when no requests are made using the session over a certain period of time. (The default session timeout is 6 hours, but it is configurable on a per-user basis.) However, termination of a session due to inactivity will not occur as long as a client application uses the API's notification facility to maintain a JMS subscription to one or more of the session's JMS notification topics. The existence of such a subscription is considered by the HMC to indicate that a client is still using the session and thus it is not terminated even if no requests are made using it. The existence of an open WebSocket associated with the API session will also prevent it from being considered inactive.

Sessions are identified by clients using a **session-id**, which is a string of up to 64 characters in length that is returned to the client in the results from a successful `Logon` operation. This string is generated in a cryptographically-secure manner. A **session-id** string is a form of authentication credentials for a user equivalent in power to a user's user ID and password. Because of this, a **session-id** should be transmitted only within SSL connections.

In order to indicate that subsequent requests are to be performed in the context of a designated session, the client supplies the appropriate **session-id** to the HMC in each such subsequent request. This is done by supplying the **session-id** as the value of the **X-API-Session** HTTP header which is an application-specific header defined by and recognized by the HMC.

The `Logon` and `Query API Version` operations are the only two operations in the Web Services API that can be performed without an API session so requests for these operations do not need to provide the **X-API-Session** HTTP header. All other operations are valid only in the context of an API session and thus requests for all other operations must supply an **X-API-Session** header with a valid **session-id** in order to be successfully executed.

Query API Version

The `Query API Version` operation returns information about the level of Web Services API supported by the HMC. This operation is supported using the BCPii interface and when used in this way returns information about the SE rather than the HMC.

HTTP method and URI

```
GET /api/version
```

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`api-major-version`	Integer	The major-version part of the API version in effect for this session
`api-minor-version`	Integer	The minor-version part of the API version in effect for this session
`hmc-version`	String (5-8)	The version number of the HMC firmware. This is a string of the form *v.r.m*, where each of *v*, *r* and *m* can be one or two digits. Example: "2.11.1". **Note:** This field is only returned for requests targeting an HMC.
`hmc-name`	String (1-16)	The name assigned to the HMC. **Note:** This field is only returned for requests targeting an HMC.
`hmc-time`	Timestamp	The current time, according to the HMC. **Note:** This field is only returned for requests targeting an HMC.
`se-version`	String (5-8)	The version number of the SE firmware. This is a string of the form *v.r.m*, where each of *v*, *r* and *m* can be one or two digits. Example: "2.15.0". **Note:** This field is only returned for requests targeting an SE.
`se-name`	String (1-8)	The name assigned to the SE. **Note:** This field is only returned for requests targeting an SE.
`se-time`	Timestamp	The current time, according to the SE. **Note:** This field is only returned for requests targeting an SE.
`classification-text`	String (1-1024)	The console's classification text. A null value is returned if the classification text is not set.

Field name	Type	Description
`vendor`	String Enum	The vendor that supplied the HMC. Non-IBM vendors are identified in an abstract fashion rather than by company name. The valid values are: • **"ibm"** - IBM • **"a"** - vendor a • **"b"** - vendor b • **"c"** - vendor c • **"d"** - vendor d • **"e"** - vendor e • **"f"** - vendor f • **"g"** - vendor g • **"h"** - vendor h • **"i"** - vendor i • **"n"** - numeric vendor value, where *n* is an integer between 0 and 255. [Added by feature **oem-hmc-ids**]
`welcome-text`	String (1-8192)	The console's welcome text. A null value is returned if the welcome text is not set.
`reflow-welcome-text`	Boolean	Indicates whether to format the welcome text to fit the width of the user's browser window.

Description

This operation returns name and version information for the HMC or SE and the API itself.

This operation can be requested without an API session being open, i.e. no **X-API-Session** header, and **session-id** is required on input.

For more information about the function included in each API version, see "Summary of API version updates" on page 7.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 113.

Under normal conditions, no error response codes are returned by this request. (HTTP Status code 500 could possibly result if internal HMC errors occur.)

Example HTTP interaction

```
GET /api/version HTTP/1.1
```

Figure 1. Query API Version: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 16 Jun 2021 15:51:46 GMT
content-type: application/json;charset=UTF-8
content-length: 225
{
   "api-major-version":4,
   "api-minor-version":1,
   "classification-text":"Top Secret",
   "hmc-name":"HMC1",
   "hmc-time":1623858706738,
   "hmc-version":"2.16.0",
   "reflow-welcome-text":true,
   "welcome-text":"Sample welcome text",
   "vendor":"ibm"
}
```

Figure 2. Query API Version: Response

Logon

The Logon operation establishes an API session with the Web Services API.

HTTP method and URI

```
POST /api/sessions
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`userid`	String	Required	The name of the HMC user to be associated with the new API session. This name may be of arbitrary length, i.e. the HMC does not have a defined maximum length.
`password`	String	Required	The password used to authenticate the HMC user identified by the **userid** field. The required length and valid characters are determined by the password policy in effect for the user ID.
`new-password`	String	Optional	A new password to be established for the user defined by the **userid** field. The required length and valid characters are determined by the password policy in effect for the user ID.
`multi-factor-authentication-code`	String (1-12)	User-dependent	The current multi-factor authentication code (time-based one-time password) used to authenticate the HMC user identified by the **userid** field. This field, or the **authentication-code** field, is required for HMC users that are configured to use HMC MFA. This field must be omitted for all other HMC users. At most one of **authentication-code**, **multi-factor-authentication-code**, or **rsa-securid-passcode** may be specified.

Field name	Type	Rqd/Opt	Description
`client-tag`	String Enum	Optional	A tag string supplied by the API client program that issued the Logon request. Valid values are: • **"mobile"** - the client asserts that it is the mobile app. This value is intended for use only by the Mobile HMC app. The last logon time for each user of this client tag is displayed in the **HMC Mobile Settings** task.
`rsa-securid-passcode`	String (1-64)	User-dependent	The current RSA SecurID passcode used to authenticate the HMC user identified by the **userid** field. This field, or the **authentication-code** field is required for certain HMC users. This field is ignored for all others. At most one of **authentication-code**, **multi-factor-authentication-code**, or **rsa-securid-passcode** may be specified.
`authentication-code`	String	User-dependent	The multi-factor authentication (MFA) token to use to authenticate the user identified by the **userid** field. This field is required for certain HMC users, optional for some, and must be omitted for all others. It can contain any supported MFA factor type. It can be used in place of **multi-factor-authentication-code** or **rsa-securid-passcode**, as appropriate. At most one of **authentication-code, multi-factor-authentication-code** or **rsa-securid-passcode** may be specified.

The largest request body accepted by this operation is 512 bytes. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`api-session`	String (1-64)	The **session-id** of the newly created session. The client must specify this value in the **X-API-Session** header of all subsequent requests that are to be performed in the session.
`notification-topic`	String (1-128)	The name of the JMS topic the HMC will use to send object-related notification messages to this session.
`job-notification-topic`	String (1-128)	The name of the JMS topic the HMC will use to send job-related notification messages to this session.
`api-major-version`	Integer	The major-version part of the API version in effect for this session
`api-minor-version`	Integer	The minor-version part of the API version in effect for this session
`password-expires`	Integer	The time interval, in days, until the user's current password expires. A value of 0 indicates that the password will expire within the next 24 hours. A value of -1 indicates that the HMC does not enforce password expiration for this user, however, if this user is authenticated with an external authentication mechanism (e.g. LDAP) such expiration might be enforced by that mechanism.

Field name	Type	Description
`shared-secret-key`	String (32)	The proposed shared secret key for the user identified in the request body. This field is only included if the user is required to establish a shared secret key; in that case, the `Logon` operation completes with HTTP status code 201 (Created).
`session-credential`	String (32)	The session-specific authentication credential for this session. This token can be used to connect to the API message broker on behalf of the HMC user associated with this session. See "Connecting to the API message broker" on page 78 for more information.

Description

This operation opens a new API session with the Web Services API. Authentication is performed as part of this process.

The characteristics and permissions of an HMC user are specified in an HMC User or User Template definition. The user name provided in the **userid** field of the request body is used to select a corresponding User or User Template based on the name. If such a User or User Template is found, the client's authority to operate as this HMC user is authenticated by validating the password provided in the **password** field using the authentication method specified in the User or User Template.

If the HMC user is configured for multi-factor authentication, additional authentication processing is required. If the HMC user currently has an established shared secret key, the user's current multi-factor authentication code provided in the **multi-factor-authentication-code** field is validated. If the HMC user does not currently have an established shared secret key, HTTP status code 201 (Created) is returned.

If the authentication described above is successful, a new API session is created and the session-id for the new session is provided in the **api-session** field in the response from this operation. This same value is also provided by an **X-API-Session** HTTP header field in the response. If all required authentication is successful, the newly created API session is fully authenticated. If HTTP status code 201 (Created) is to be returned, a partially-authenticated API session is created. In this case an `Establish Shared Secret Key` operation, using the key supplied in the **shared-secret-key** field in the response body, must be issued to establish a shared secret key and complete the API logon sequence, thereby converting the partially-authenticated session into a fully-authenticated session. The operations available to a partially-authenticated session are limited to the `Establish Shared Secret Key` and `Logoff` operations.

If the request specifies an **X-API-Session** HTTP header field on input (indicating that this operation be performed under some designated session), the logon request fails and status code 400 (Bad Request) is returned.

If an HMC User or User Template corresponding to the user ID field does not exist, or if the password or multi-factor authentication code validation fails, the logon request fails and status code 403 (Forbidden) is returned. There is no reason code to distinguish these reasons for the failure. If the User or User Template is marked as disabled or the associated password has expired, or if the User or User Template is not configured to allow use of the API, the logon request also fails with status code 403 (Forbidden) and a reason code identifying the specific cause.

If all required user authentication is successful and the request body contains the optional **new-password** field, the password associated with the user is changed to the specified new value as part of the `Logon` operation. If the new password does not meet the requirements of the password policy in effect for this user or if the password is not changeable because it is managed by an external authentication mechanism, the request fails with status code 400 (Bad Request) and a reason code indicating the cause of the failure.

As part of establishing the new API session, names are assigned for the JMS topics that will be used by the HMC to send object-related and job-related notification messages to this session and the names of these topics are provided in fields of the response body. The name of the topic used for object-related notifications is provided in the **notification-topic** field of the response, and the name of the topic used for job-related notifications is provided in the **job-notification-topic** field.

Authorization requirements

This operation has the following authorization requirement:

- The HMC User Profile or User Template selected by the **userid** field must be configured to allow use of the Web Services API.

HTTP status and reason codes

On success, HTTP status code 200 (OK) or 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 116.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 50. Logon: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	12	The request specified an **X-API-Session** header, which is interpreted as an attempt to unnecessarily logon again when already logged on.
	13	The maximum number of logged in user sessions for this user ID has been reached; no more are allowed.
	43	The password for this user cannot be changed, for example because it is managed in an external authentication mechanism such as LDAP.
	44	The new password does not conform to the requirements of the password policy in effect for this user.
	45	The user's password has expired and no **new-password** field was specified. This reason code is only applicable for users who are not required to use MFA.
	49	The specified multi-factor authentication code is correct, but it has already been used. A code may only be used once. Wait until a new code is available and try again.
403 (Forbidden)	0	Login failed. Try the operation again. If the problem persists, contact your security administrator.
	40	The user is disabled.
	41	The user is not authorized to use the HMC Web Services interface.

Table 50. Logon: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	50	The MFA Server requires more information in order to authenticate the API user. Details are provided in a JSON object in the response body. See the MFA server documentation for a description of that JSON object and the format of the JSON object required to contain the additional information. That additional information must be provided through the `Provide Requested MFA Information` operation. The **error-details** field of the response body contains an mfa-info-request object containing additional details, found in Table 51 on page 119.
	51	The user's password has expired. A new password must be set through the `Change Logon Password` operation. The **error-details** field of the response body contains a change-password-info object containing additional information, described in Table 52 on page 119. This code is only applicable for users who are required to use MFA.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

On completion where the HTTP status code is 409 (Conflict) with reason code 50, the standard error response body contains an error-details field that contains a description of the additional information requested by the MFA server. The value of the **error-details** field is a nested mfa-info-request object with the following fields:

Table 51. mfa-info-request nested object properties		
Field name	**Type**	**Description**
`resume-id`	String (4-128)	A string which identifies the authentication request to which this response applies.
`api-session`	String (1-64)	The **session-id** of the partially-authenticated API session to which this response applies.
`information-request`	Object	A JSON object describing the additional information requested by the MFA server. See the MFA server documentation for details.

On completion where the HTTP status code is 409 (Conflict) with reason code 51, the standard error response body contains an error-details field that contains information for use on the required **Change Logon Password** operation. The value of the **error-details** field is a nested change-password-info object with the following field:

Table 52. change-password-info nested object properties		
Field name	**Type**	**Description**
`api-session`	String (1-64)	The **session-id** of the partially-authenticated API session to which this response applies.

Usage notes

- The Logon operation checks for and prevents requests that specify an **X-API-Session** header on input in order to detect client applications that unnecessarily log on again when already logged on. It is valid to have multiple sessions, but in order to more explicitly indicate that this is desired, the client application needs to request each logon without referencing any existing session.

- Some of the information returned by this operation is also present in the response body of a successful `Get Notification Topics` request. Specifically, the information contained in the **notification-topic** and **job-notification-topic** fields is also included in the `Get Notification Topics` response. That operation identifies all JMS topics available to the API user, possibly including topics other than those identified in the `Logon` response.
- The MFA (multi-factor authentication) code provided in the **authentication-code** field can be any type of MFA token supported by the HMC, including those that can be specified in the **multi-factor-authentication-code** and **rsa-securid-passcode** fields. Only one of these three fields should be included, and it should only be included when required. The exact format and content of the authentication code depends on the specific type of factor being supplied, and it may vary according to configuration parameters controlled by the security administrator. See the **Logon** task on the console help system for information on the types of factors supported by the HMC.
- The MFA code provided in the **rsa-securid-passcode** field is the API user's current RSA SecurID passcode. It is typically a 6- or 8-digit number and may also include a PIN. The exact format and content of the passcode is configured by the RSA authentication server administrator. This passcode is only required for users configured to use IBM Z® MFA and the RSA SecurID factor type. An RSA SecurID passcode can be specified in either the **rsa-securid-passcode** field or the **authentication-code** field.
- The MFA code provided in the **multi-factor-authentication-code** field for a user configured to use HMC MFA is a time-based one-time password as defined in RFC 6238, *TOTP: Time-Based One-Time Password Algorithm*, May 2011, (available at: https://tools.ietf.org/html/rfc6238 from the Internet Engineering Task Force). The TOTP algorithm uses a shared secret key and the current time of day to calculate the TOTP for the current 30-second interval. The TOTP is a 6-digit number. A TOTP can be specified in either the **multi-factor-authentication-code** field or the **authentication-code** field.
- When using HMC MFA the user must establish a shared secret key. This key is sensitive security information much like a password and is to be known only by the user and the HMC. This key can be established through the `Establish Shared Secret Key` operation or by logging on to the HMC through the local GUI interface or a remote web browser. The HMC presents this key to the user only once (during the first logon after being required to use HMC MFA or having their shared secret key be invalidated by an administrator), and it is the user's responsibility to have it available for use during subsequent API and GUI logons, which will require the user's current multi-factor authentication code.
- The normal sequence of operations to establish a shared secret key through the APIs is as follows:
  - Issue a `Logon` operation with a valid logon password.
  - The `Logon` completes with HTTP status code 201 (Created) and returns an API session ID for a partially-authenticated session and a proposed shared secret key.
  - Issue an `Establish Shared Secret Key` operation with the API session ID and the current TOTP calculated using the proposed shared secret key.
  - The `Establish Shared Secret Key` operation completes with HTTP status code 204 (No Content), the partially-authenticated session is converted into a fully-authenticated session and the proposed key is now the user's officially established shared secret key.
- When using HMC MFA it is important for the API client's time of day clock to be reasonably in sync with the HMC's clock, because the current time of day is used when calculating the user's current multi-factor authentication code. To determine if the clocks are reasonably in sync, the `Query API Version` operation may be used to obtain the HMC's current time, which can then be compared to that of the API client.
- If an `Establish Shared Secret Key` operation is required to complete a logon sequence that includes a **client-tag** field, the last mobile app logon time for the API user is not updated until the `Establish Shared Secret Key` operation completes successfully.

Example HTTP interaction

```
POST /api/sessions HTTP/1.1
content-type: application/json
content-length: 58
{
   "password": "12345678",
   "userid": "APIUSER"
}
```

Figure 3. Logon: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Wed, 02 Aug 2017 18:41:27 GMT
x-api-session: 4hy7c4nogldz4b59ajegzb1dulec641ziyv6uf73zs43205edv
content-type: application/json;charset=UTF-8
content-length: 281
{
   "api-major-version": 20,
   "api-minor-version": 2,
   "api-session": "4hy7c4nogldz4b59ajegzb1dulec641ziyv6uf73zs43205edv",
   "job-notification-topic": "APIUSER.229job",
   "notification-topic": "APIUSER.229",
   "password-expires": 29,
   "session-credential": "un8bu462g37aw9j0o8pltontz3szt35jh4b1qe2toxt6fkhl4"
}
```

Figure 4. Logon: Response

Establish Shared Secret Key

The `Establish Shared Secret Key` operation completes the authentication of a partially-authenticated API session and establishes the user's multi-factor authentication shared secret key.

HTTP method and URI

```
POST /api/sessions/operations/establish-shared-secret-key
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`multi-factor-authentication-code`	String (1-12)	Required	The current multi-factor authentication code (time-based one-time password) to be used to authenticate the HMC user associated with the partially-authenticated API session specified on the **X-API-Session** request header.

Description

This operation establishes a shared secret key for a user that is configured for multi-factor authentication. It completes the authentication of a partially-authenticated API session created by a previous `Logon` operation that completed with HTTP status code 201 (Created). That API session's ID must be specified on the **X-API-Session** request header. The request body must contain the current multi-factor authentication code calculated using the proposed shared secret key returned in the response body of the aforementioned `Logon` operation.

If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the multi-factor authentication code is not correct for the proposed shared secret key and current time of day, HTTP status code 403 (Forbidden) is returned. If the requirement for the user to establish a shared secret key no longer exists, HTTP status code 409 (Conflict) is returned with a reason code that indicates what has changed.

If the operation does not complete with status code 204 (No Content) or status code 409 (Conflict) with reason code 51, the partially-authenticated session is destroyed.

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of the partially-authenticated API session and the current multi-factor authentication code for that session.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	46	The **X-API-Session** request header does not identify a partially-authenticated session.
403 (Forbidden)	0	Login failed. Try the operation again. If the problem persists, contact your security administrator.
409 (Conflict	12	The user associated with the API session identified by the **X-API-Session** request header already has an established shared secret key. That key could have been established by another API session or a GUI logon.
	51	The user's password has expired. A new password must be set through the `Change Logon Password` operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage notes

See the usage notes for the `Logon` operation for more information on using multi-factor authentication with the APIs.

Example HTTP interaction

```
POST /api/sessions/operations/establish-shared-secret-key HTTP/1.1
x-api-session: 5ql67thiw2og8ysixzljv8pwwmb4exfp85h9lu23a2irjxaq0w
content-type: application/json
content-length: 46
{
   "multi-factor-authentication-code":"314159"
}
```

Figure 5. Establish Shared Secret Key: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 16 Dec 2016 21:21:09 GMT

<No response body>
```

Figure 6. Establish Shared Secret Key: Response

Provide Requested MFA Information

The `Provide Requested MFA Information` operation satisfies a request from an MFA server for more user authentication information.

HTTP method and URI

```
POST /api/sessions/operations/provide-more-mfa-information
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`resume-id`	String (4-128)	Required	A string which identifies the authentication request to which this operation applies.
`requested-information`	Object	Required	A JSON object containing the requested information in the format required by the MFA server.

Response body contents

On successful completion, the response body contains a JSON object with the same format as the response body for a `Logon` operation that completed with HTTP status code 200 (OK). See the `Logon` "Response body contents" on page 116.

Description

This operation supplies the additional user authentication information requested by an MFA server during a previous authentication attempt. For example, the API user may be required to change their MFA PIN or provide their next MFA token code. The previous attempt could have been through a `Logon` operation or a `Provide Requested MFA Information` operation.

If the provided information is successfully validated, the MFA server may require further information. In that case, HTTP status code 409 (Conflict) with reason code 50 is returned, and the response body

describes the additional required information. The **error-details** field of the response body contains an mfa-info-request object described in Table 51 on page 119.

If the provided information is successfully validated and the MFA server requires no further information, this operation completes the MFA authentication of a partially-authenticated API session created by a previous `Logon` operation that completed with HTTP status code 409 (Conflict) with reason code 50. That partially-authenticated API session's ID must be specified on the **X-API-Session** request header. If the user's logon password has expired, HTTP status code 409 (Conflict) with reason code 51 is returned, and the password must be changed through the `Change Logon Password` operation to complete the logon sequence.

The request body is validated against the schema described in "Request body contents" on page 123. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body must contain the **resume-id** returned in the response from a `Logon` or prior `Provide Requested MFA Information` operation as well as the information specifically requested by that response. If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the MFA server did not authenticate the credentials for any reason, HTTP status code 403 (Forbidden) is returned and additional error details may be available.

If the operation does not complete with status code 200 (OK) or status code 409 (Conflict) with reason code 50 or 51, the partially-authenticated session is destroyed.

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of the partially-authenticated API session and the **resume-id** from a prior request for that API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, the response body is provided as described in "Response body contents" on page 123.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 53. Provide Requested MFA Information: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	40	The **resume-id** does not identify the currently outstanding request for additional information for the specified API session.
	46	The **X-API-Session** request header does not identify a partially-authenticated session that requires more MFA information.
403 (Forbidden)	0	Login failed. Try the operation again. If the problem persists, contact your security administrator.
	40	The MFA server failed to authenticate the credentials or requires the user to re-authenticate. The **error-details** field of the response body contains an mfa-requested-information-failures object. The mfa-requested-information-failures object is described in the next table.

Table 53. Provide Requested MFA Information: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict	50	The MFA Server requires more information in order to authenticate the API user. Details are provided in a JSON object in the response body. See the MFA server documentation for a description of that JSON object and the format of the JSON object required to contain the additional information. That additional information must be provided through the `Provide Requested MFA Information` operation.
	51	The user's password has expired. The `Change Logon Password` operation must be issued to change the password and continue the logon sequence, possibly completing it.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Table 54. mfa-requested-information-failures object		
Field name	**Type**	**Description**
`failing-factors`	Array of mfa-requested-information-failing-factor objects	The array of objects describing the failures, one per failing factor, as returned by the MFA server.

Table 55. mfa-requested-information-failing-factor object		
Field name	**Type**	**Description**
`name`	String	The name/key of the failing factor
`reason`	String	The reason code of the failure, in the form of *xx.yy*, as returned by the MFA server.
`message`	String	The message describing the failure, as returned by the MFA server. May be an empty string.

Example HTTP interaction

```
POST /api/sessions/operations/provide-more-mfa-information HTTP/1.1
x-api-session: 12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3
content-type: application/json
content-length: 214


{
   "resume-id":"TgJI8gbkFYveL+DXToSM79gwI5Dx+Vs",
   "requested-information":{
      "resumeID":"TgJI8gbkFYveL+DXToSM79gwI5Dx+Vs",
      "apiVersion":2,
      "factors":[
         {
            "factorName":"AZFSIDP1",
            "credentialObject":{
               "passCode":"12345678"
            }
         }
      ]
   }
}
```

Figure 7. Provide Requested MFA Information: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 01 Nov 2019 16:33:15 GMT
x-api-session: 12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3
content-type: application/json
content-length: 301
{
  "api-major-version":3,
  "api-minor-version":2,
  "api-session":"12i0xmvsej62fjwwhryd7wntf671t1wdt0qd5geh9dalwijsa3",
  "job-notification-topic":"apiuser1.13job",
  "notification-topic":"apiuser1.13",
  "password-expires":28,
  "session-credential":"3q2dbw5xbrzgjzcnn1r13shuv3k00vcy1gjhqtfg2fogpor6aa"
}
```

Figure 8. Provide Requested MFA Information: Response

Usage notes

See the usage notes for the `Logon` operation for more information on using multi-factor authentication with the APIs.

Change Logon Password

The `Change Logon Password` operation changes the logon password during a logon sequence for an MFA-enabled user.

HTTP method and URI

```
POST /api/sessions/operations/change-logon-password
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`current-password`	String	Required	The password used to authenticate the HMC user associated with the partially-authenticated API session identified by the **X-API-Session** HTTP request header.
`new-password`	String	Required	A new password to be established for the user. The required length and valid characters are determined by the password policy in effect for the user ID.

Response body contents

On successful completion, the response body contains a JSON object with the same format as the response body for a Logon operation that completed with HTTP status code 200 (OK). See the Logon "Response body contents" on page 116.

Description

This operation changes the password used to authenticate the user associated with a partially-authenticated API session when that user's password has expired. The API session is identified by the **X-API-Session** HTTP request header. If the new password does not meet the requirements of the password policy in effect for this user or if the password is not changeable because it is managed by an external authentication mechanism, the request fails with status code 400 (Bad Request) and a reason code indicating the cause of the failure.

Upon successful completion the user's password is changed and this operation completes the authentication of a partially-authenticated API session created by a previous Logon operation that completed with HTTP status code 201 (Created) or HTTP status code 409 (Conflict) with reason code 50 or 51. That partially-authenticated API session's ID must be specified on the **X-API-Session** request header.

The request body is validated against the schema described in "Request body contents" on page 126. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the **X-API-Session** request header does not identify a partially-authenticated session, HTTP status code 400 (Bad Request) is returned. If the specified current password is not correct, HTTP status code 403 (Forbidden) is returned.

If the operation does not complete with status code 200 (OK), the partially-authenticated session identified by the **X-API-Session** HTTP request header is destroyed.

Authorization requirements

The request must contain the session ID of a partially-authenticated API session for a user whose logon password has expired and the current password for the user associated with that API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, the response body is provided as described in "Response body contents" on page 127.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 56. Change Logon Password: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	44	The new password does not conform to the requirements of the password policy in effect for this user.
	46	The **X-API-Session** request header does not identify a partially-authenticated session that requires a logon password change.
403 (Forbidden)	0	Login failed. Try the operation again. If the problem persists, contact your security administrator.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/change-logon-password HTTP/1.1
x-api-session: 4uyp0d2o7q0495p4ktmw9mt4ef4w2owsrqjjgch7rbhds17u73
content-type: application/json
content-length: 60
{
   "current-password":"12345678",
   "new-password":"87654321"
}
```

Figure 9. Change Logon Password: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Sep 2019 19:51:25 GMT
x-api-session: 18l8cbyrsndoabvzdy1gq6oscqd77hjmsyy4j4ezvf0tptf5vp
content-type: application/json
content-length: 301
{
  "api-major-version":3,
  "api-minor-version":2,
  "api-session":"18l8cbyrsndoabvzdy1gq6oscqd77hjmsyy4j4ezvf0tptf5vp",
  "job-notification-topic":"apiuser1.12job",
  "notification-topic":"apiuser1.12",
  "password-expires":12,
  "session-credential":"2q2dbw5xbrzgjzcnn1r13shuv3k00vcy1gjhqtfg2fogpor6ii"
}
```

Figure 10. Change Logon Password: Response

Usage notes

See the usage notes for the Logon operation for more information on using multi-factor authentication with the APIs.

Verify Logon Password

The `Verify Logon Password` operation verifies that the logon password for the user's session is correct.

HTTP method and URI

```
POST /api/sessions/operations/verify-logon-password
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`password`	String	Required	The password of the HMC user associated with the authenticated API session identified by the **X-API-Session** HTTP request header.

Description

This operation verifies the password matches the password of the HMC user associated with the authenticated API session identified by the **X-API-Session** HTTP request header.

If the password is correct, HTTP status code 204 (No Content) is returned. If the password is not correct, status code 400 (Bad Request) is returned.

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of a fully-authenticated API session and the current logon password for that session.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 57. Verify Logon Password: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	41	The password in the request body is not correct or could not be verified.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/verify-logon-password HTTP/1.1
x-api-session: 1r8f5vemenfnhilj5yeto2y6buybts5t09a61769zpq4rmb4g2
Content-Type: application/json
Content-Length: 24
{
  "password":"SomePass"
}
```

Figure 11. Verify Logon Password: Request

```
204
<No response body>
```

Figure 12. Verify Logon Password: Response

Usage note

This operation allows a client with an active authenticated API session to re-check the user's password. This could be useful for disruptive action confirmations such as the HMC user interface provides.

Logoff

The `Logoff` operation closes an API session with the Web Services API.

HTTP method and URI

```
DELETE /api/sessions/this-session
```

Description

This operation closes an API session with the Web Services API.

The session to be closed is indicated by the **session-id** in the **X-API-Session** header of the request. If the **session-id** designates an open session, the API session is closed and status code 204 (No Content) is returned. Closing of the API session includes closing/deleting any Metrics Service retrieval contexts, JMS notification topics, and SSE streams associated with the session. However, asynchronous actions initiated by the session continue to run.

Once a session is closed, its **session-id** is no longer valid for use in subsequent Web Services API requests. Attempts to do so will result in the same errors as any other attempt to use a session-requiring operation without providing a valid **session-id**.

Authorization requirements

This operation has the following authorization requirement:

- No explicit authorization is required, however the client application must possess and present a valid **session-id** of the session to be closed.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/sessions/this-session HTTP/1.1
x-api-session: zkspmapxgtcasy5uixmtwuaudqe8ha6fy0006bzmxsm2bd8yo
```

Figure 13. Logoff: Request

```
204 No Content
date: Wed, 20 Jul 2011 18:33:56 GMT
x-request-id: Sx32 Rx0
server: zSeries management console API web server / 1.0
cache-control: no-cache
```

Figure 14. Logoff: Response

Get Notification Topics

The `Get Notification Topics` operation returns a structure that describes the JMS notification topics associated with the API session. These topics allow the user to receive various types of asynchronous notifications from the HMC.

HTTP method and URI

```
GET /api/sessions/operations/get-notification-topics
```

Response body contents

On successful completion, the response body is a JSON object with the following field:

Field name	Type	Description
`topics`	Array of objects	Array of nested topic-info objects as described in the next table

Each nested topic-info object contains the following fields:

Table 58. topic-info object		
Field name	**Type**	**Description**
`topic-type`	String Enum	The type of notification topic, which provides an indication of the type of data found on the topic. Except for **os-message-notification**, a given value will only be represented at most once within a single response. One of the following values: • **"object-notification"** - The object notification topic. This topic is consistent with the information returned in the **notification-topic** field in the response body of a successful Logon request. It is used by the HMC to send object-related notifications to this session. • **"job-notification"**- The job notification topic. This topic is consistent with the information returned in the **job-notification-topic** field in the response body of a successful Logon request. It is used by the HMC to send job-related notifications to this session. • **"audit-notification"** - the audit notification topic. This topic is used by the HMC to send audit-related events to this session. • **"security-notification"** - the security notification topic. This topic is used by the HMC to send security-related events to this session. • **"os-message-notification"** - an operating system message notification topic. Topics of this type are used by the HMC to send notifications that pertain to new or refreshed messages generated by the operating system running in a partition. More than one of these might exist for this session. Additional fields specific to this topic type are described later in this table. • **"console-notification"** - the console notification topic. This topic is used to send console events to this session. • **"disabled-wait"** - the disabled wait notification topic. This topic is used to send disabled wait events to this session. • **"capacity-record-change"** - the capacity record change topic. This topic is used to send capacity record change events to this session. • **"capacity-change"** - the capacity change topic. This topic is used to send capacity change events to this session. [Updated by feature **bcpii-notifications**]
`topic-name`	String (1-128)	The name of the notification topic. API users can connect using this name to receive notifications for the topic.
`object-uri`	String/ URI	When the **topic-type** is **"os-message-notification"**, this field is the canonical URI path of the partition object for which this topic exists. This field does not exist for the other topic types.
`include-refresh-messages`	Boolean	When the **topic-type** is **"os-message-notification"**, this field indicates whether refresh operating system messages will be sent on this topic. A value of **true** indicates that refresh messages will be sent. A value of **false** indicates that no refresh messages will be sent. This field does not exist for the other topic types.

Description

This operation returns a list of all JMS topics to which the API user is authorized to connect. As there exists at least one JMS topic available to any authenticated user, the returned JSON array will never be empty.

Authorization Requirement

This operation has the following authorization requirement:

- No explicit authorization is required; however, the response to this request is limited to the topics to which the user is authorized to connect.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 131.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/sessions/operations/get-notification-topics HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4qymqa8ktzjep7dbyrll0k
```

Figure 15. Get Notification Topics: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json;charset=UTF-8
{ "topics" :
  [
    { "topic-type":"object-notification", "topic-name":"mikeuser.1" },
    { "topic-type":"job-notification", "topic-name":"mikeuser.1job" },
    { "topic-type":"audit-notification", "topic-name":"mikeuser.1aud" },
    { "topic-type":"security-notification", "topic-name":"mikeuser.1sec" }
    { "topic-type": "os-message-notification",
      "topic-name": "mikeuser.1osmsg.cpc1.lpar1",
      "object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
      "include-refresh-messages": true },
    { "topic-type": "os-message-notification",
      "topic-name": "mikeuser.1osmsg.cpc2.par7",
      "object-uri": "/api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd",
      "include-refresh-messages": false }
  ]
}
```

Figure 16. Get Notification Topics: Response

Usage notes

Some of the information returned by this operation is also present in the response body of a successful `Logon` request. This operation is intended to provide a superset and will contain all JMS topics available to the API user including the two topics indicated in the `Logon` response.

Create Server-Sent Events Stream

The `Create Server-Sent Events Stream` operation creates an SSE stream with the registration values passed in the request body. The returned **stream-id** can then be used to listen for events on the stream with `Open Server-Sent Events Stream` or reconfigured with `Update Server-Sent Events Stream`.

HTTP method and URI

```
POST /api/sessions/operations/create-server-sent-events-stream
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`event-names`	Array of String Enum	Required	The names of the events to register for on the stream. Clients can match the values here to the name property of the SSE event to determine what type of event is being received. Zero or more of the following must be included: • **"status-change"** – include Status Change events. An optional **status-change-filter** field can be included to limit what status events are emitted. • **"property-change"** – include Property Change events. An optional **property-change-filter** field can be included to limit what property events are emitted. • **"inventory-change"** – include Inventory Change events. An optional **inventory-change-filter** field can be included to limit what inventory events are emitted • **"job-completion"** – include completion events of jobs created by this session. • **"audit-log-entry"** – include events of new entries to the HMC's audit log. This requires permission to the **Audit and Log Management** task. • **"security-log-entry"** – include events of new entries to the HMC's security log. This requires permission to the **View Security Logs** task. • **"console-event-log-entry"** – include events of new entries to the HMC's event log. This requires permission to the **View Console Events** task. • **"os-message"** – include events of new OS messages on registered objects. This requires permission to the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode. When this event name is included, specifying for which objects is required with the **os-messages-filter** field. An empty array will result in no events being sent to the stream.
`status-change-filter`	object-change-filter object	Optional	The object-change-filter object, as described in Table 59 on page 136, to specify the objects and/or elements on which to register for Status Change events. If omitted, registration will be on all objects and elements that support Status Change events. It is ignored if **"status-change"** is not included in **event-names**

Field name	Type	Rqd/Opt	Description
`property-change-filters`	Array of property-change-filter objects	Optional	An array of property-change-filter objects, as described in Table 60 on page 137, to specify the properties, objects, and/or elements on which to register for Property Change events. Providing multiple filter objects allows registering for type-specific properties on different types of objects. An event need only pass one filter to be sent. The minimum array size is 1. If omitted, registration will be on all properties of all objects and elements that support Property Change events. It is ignored if **"property-change"** is not included in **event-names**.
`inventory-change-filter`	object-change-filter object	Optional	The object-change-filter object, as described below, to specify the objects and/or elements on which to register for Inventory Change events. If omitted, registration will be on all objects and elements that support Inventory Change events. It is ignored if **"inventory-change"** is not included in **event-names**.
`os-message-filter`	os-message-filter object	Optional*	The os-message-filter object, as described in Table 61 on page 137, to specify the objects on which to register for OS Message events. * - This is required if **"os-message"** is included in **event-names**. Otherwise, it is ignored.

Each nested object-change-filter object contains the following fields:

Table 59. object-change-filter nested object

Field name	Type	Rqd/Opt	Description
`objects`	Array of String/ Object URI Pattern	Optional	Allows matching on any number of objects by their **object-uri** property. Each value must be in the form of an Object URI Pattern. The minimum array size is 1.
`elements`	Array of String/ Element URI Pattern	Optional	Allows matching on any number of elements by their **element-uri** property. Each value must be in the form of an Element URI Pattern. The minimum array size is 1.

At least one field must be included in the object-change-filter object. Multiple fields can be specified, and an event needs to match on at least one to be sent.

Each nested property-change-filter object contains the same fields as object-change-filter objects, with the addition of the following field:

Table 60. property-change-filter nested object			
Field name	**Type**	**Rqd/Opt**	**Description**
`properties`	Array of String	Optional	Allows matching on any number of property names. Each value must be a property name. The minimum array size is 1.

At least one field must be included in the property-change-filter object. For a Property Change event to be sent, its object or element must match at least one of the object filter fields, if supplied, and match the properties field, if supplied.

Each nested os-message-filter object contains the following field:

Table 61. os-message-filter nested object			
Field name	**Type**	**Rqd/Opt**	**Description**
`objects`	Array of String/ Object URI Pattern	Required	Allows matching on any number of objects by their **object-uri** property. Each value must be in the form of an Object URI Pattern, except the **object-classification** must be specified and describe objects that support operating system messages. These are objects with a classification of **"partitions"** or **"logical-partitions"**. The minimum array size is 1.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`stream-id`	String	The unique identifier of the created stream.

Description

This operation creates an SSE stream for the session that can then be used to listen for asynchronous event notifications using an SSE client with `Open Server-Sent Events Stream`. Refer to "Server-Sent Events (SSE)" on page 90 for more details.

Note that the **stream-id** is used to authorize the `Open Server-Sent Events Stream` operation. Clients should therefore handle it with the same care as the **session-id** authorization token from the **Logon** operation.

On successful execution, a stream has been created with the passed configuration, its **stream-id** is provided in the response body, and HTTP status code 200 (OK) is returned.

All object and element URI patterns are verified to be syntactically valid. However, the existence or permission to such objects and elements is not verified. Events for such objects and elements would, of course, never be emitted to an unauthorized user.

If an event name is included in the request body and the user does not have the required action/task permission for it, 403 (Forbidden) is returned.

Clients are encouraged to register for as much as possible in a single stream. However, multiple streams are supported up to a limit of 50 per session. If another is attempted to be created at that point, 503 (Service Unavailable) is returned.

Authorization Requirement

This operation has no general authorization requirements. However, the following requirements apply to specific configurations:

- The **"audit-log-entry"** event name requires action/task permission to the **Audit and Log Management** task.
- The **"security-log-entry"** event name requires action/task permission to the **View Security Logs** task.
- The **"console-event-log-entry"** event name requires action/task permission to the **View Console Events** task.
- The **"os-message"** event name requires action/task permission to the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 137.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
503 (Service Unavailable)	5	The maximum number of streams for the session has been reached (50).

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/create-server-sent-events-stream HTTP/1.1
x-api-session: vpqcnuhhg05s8s2euvh4mjj57nbikuluqf6bbzuk4vb5swaxr
Content-Type: application/json
Content-Length: 36
{
   "event-names":[
      "property-change"
   ]
}
```

Figure 17. Create Server-Sent Events Stream: Request

```
200
Content-Type: application/json
Content-Length: 66
{
   "stream-id":"5fhrqbujxxdkavwfe4v1hl5o6gwf4iookr318ycnrjy21bb8t2"
}
```

Figure 18. Create Server-Sent Events Stream: Response

Update Server-Sent Events Stream

The `Update Server-Sent Events Stream` operation updates an existing SSE stream with the registration values passed in the request body.

HTTP method and URI

```
POST /api/sessions/operations/update-server-sent-events-stream
```

Request body contents

The request body is expected to contain a JSON object identical to the `Create Server-Sent Events Stream`'s "Request body contents" on page 134 with the following additional field:

Field name	Type	Rqd/Opt	Description
`stream-id`	String	Required	The unique identifier of the stream whose configuration is to be updated.

Description

This operation updates an existing SSE stream configuration to the values passed in the request body. It can be executed regardless of the stream currently being open or closed. Refer to "Server-Sent Events (SSE)" on page 90 for more details.

On successful execution, a stream has been updated with the passed configuration and HTTP status code 204 (No Content) is returned. If the stream is currently open with initial events requested, then any new initial events required based on the new configuration will be emitted asynchronous to this operation.

All object and element URI patterns are verified to be syntactically valid. However, the existence or permission to such objects and elements is not verified. Events for such objects and elements would, of course, never be emitted to an unauthorized user.

If an event name is included in the request body and the user does not have the required action/task permission for it, status code 403 (Forbidden) is returned. If the passed **stream-id** does not identify a known stream of the session, 404 (Not Found) is returned.

Authorization Requirement

This operation has no general authorization requirements. However, the following requirements apply to specific configurations:

- The **"audit-log-entry"** event name requires action/task permission to the **Audit and Log Management** task.
- The **"security-log-entry"** event name requires action/task permission to the **View Security Logs** task.
- The **"console-event-log-entry"** event name requires action/task permission to the **View Console Events** task.
- The **"os-message"** event name requires action/task permission to the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	7	The **stream-id** does not designate a known stream for the session.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/update-server-sent-events-stream HTTP/1.1
x-api-session: vpqcnuhhg05s8s2euvh4mjj57nbikuluqf6bbzuk4vb5swaxr
Content-Type: application/json
Content-Length: 120
{
   "event-names":[
      "status-change",
      "property-change"
   ],
   "stream-id":"5fhrqbujxxdkavwfe4v1hl5o6gwf4iookr318ycnrjy21bb8t2"
}
```

Figure 19. Update Server-Sent Events Stream: Request

```
204

<No response body>
```

Figure 20. Update Server-Sent Events Stream: Response

Delete Server-Sent Events Stream

The `Delete Server-Sent Events Stream` operation deletes an existing SSE stream.

HTTP method and URI

```
POST /api/sessions/operations/delete-server-sent-events-stream
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`stream-id`	String	Required	The unique identifier of the stream whose configuration is to be deleted.

Description

This operation deletes an SSE stream configuration. If the stream is currently open, it is closed prior to being deleted. Refer to “Server-Sent Events (SSE)” on page 90 for more details.

On successful execution, the identified stream has been deleted and HTTP status code 204 (No Content) is returned. If the passed **stream-id** does not identify a known stream of the session, status code 404 (Not Found) is returned.

Authorization Requirement

The only authorization requirement is the stream must be owned by the requesting session.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	7	The **stream-id** does not designate a known stream for the session.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/delete-server-sent-events-stream HTTP/1.1
x-api-session: vpqcnuhhg05s8s2euvh4mjj57nbikuluqf6bbzuk4vb5swaxr
Content-Type: application/json
Content-Length: 67
{
   "stream-id":"5fhrqbujxxdkavwfe4v1hl5o6gwf4iookr318ycnrjy21bb8t2"
}
```

Figure 21. Delete Server-Sent Events Stream: Request

```
204

<No response body>
```

Figure 22. Delete Server-Sent Events Stream: Response

Open Server-Sent Events Stream

`Open Server-Sent Events Stream` is not a WSAPI operation in the standard sense, but rather the URI used by an SSE client to open a stream previously created by the `Create Server-Sent Events Stream` operation. The stream identifier is passed as a query parameter to the URI.

HTTP method and URI

```
GET /api/sessions/operations/open-server-sent-events-stream
```

Query parameters:

Name	Type	Rqd/Opt	Description
`stream-id`	String	Required	The configured stream identifier to subscribe the SSE client to, as returned from the `Create Server-Sent Events Stream` operation.
`initial-events`	Boolean	Optional	A value of **true** causes an initial set of events to be sent on the stream for everything registered for. A value of **false** causes only asynchronous notifications of changes to be sent. The default is **false**.

Response body contents

On successful completion, the response body will be as defined by the SSE specification. Events may then be emitted to the stream in the form described by "Notification event formats" on page 91.

Description

This operation opens a configured stream and registers a client to begin listening for events. Refer to "Server-Sent Events (SSE)" on page 90 for more details.

On successful execution, response code 200 (OK) is returned with **Content-**Type **"text/event-stream"** and events may begin flowing. The passed **stream-id** must be a valid, configured identifier of a stream belonging to a fully authenticated session; otherwise, status code 404 (Not Found) is returned. If the identified stream is currently open, it is closed prior to being re-opened.

If initial events are requested, they will be processed and delivered asynchronously to this request with the special events of "initial-events-start" and "initial-events-end" before and after them, respectively. Initial events are guaranteed to be delivered prior to any change events, although any change events that occur during initial event processing will still be delivered once that completes.

Note also that the request of initial events will be remembered for the life of the stream being open. Therefore, if the stream's configuration is updated while open to add additional registrations, a set of initial events will be sent for the added items. Additionally, if any object-related registrations are filtered by class or not filtered at all, new objects in the inventory will cause initial events for them.

Authorization Requirement

This operation has no authorization requirements.

The **stream-id** is the authorization token for the operation, created by a fully authenticated session with the `Create Server-Sent Events Stream` operation, ensuring the stream is only sent events the user is authorized to receive.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned with **Content-**Type **"text/event-stream"**.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	7	The **stream-id** does not designate a known stream for the session.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Usage notes

The intended use is to simply pass the URI with the **stream-id** as a query parameter to an SSE client’s event source implementation which is then opened, such as follows:

```
new EventSource(
    https://myhmc:6794/api/sessions/operations/open-server-sent-events-stream?stream-
id=mystreamid
)
```

Get Server-Sent Events Stream Last Event ID

`Get Server-Sent Events Stream Last Event ID` operation returns the last sent event ID of an SSE stream. [Added by feature **mobile-enhanced-push**]

HTTP method and URI

```
POST /api/sessions/operations/get-server-sent-events-stream-last-event-id
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Name	Type	Rqd/Opt	Description
`stream-id`	String	Required	The unique identifier of the stream from which to get the last sent event ID.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Name	Type	Description
`id`	String	The unique identifier of the stream's last sent event.

Description

This operation returns the event **id** of the last sent event of an SSE stream. If no event has been sent on the stream, -1 is returned. This operation can be used by clients with unstable connections. If a connection is lost and regained, the last sent event **id** can be checked to know if any events were missed while it was disconnected.

Refer to “Server-Sent Events (SSE)” on page 90 for more details.

On successful execution, the identified stream's last sent event **id** is provided in the response body and HTTP status code 200 (OK) is returned. If the passed **stream-id** does not identify a known stream of the session, status code 404 (Not Found) is returned.

Authorization Requirement

The only authorization requirement is the stream must be owned by the requesting session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 143.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	7	The **stream-id** does not designate a known stream for the session.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/sessions/operations/get-server-sent-events-stream-last-event-id HTTP/1.1
x-api-session: 408qm0glnlaijlp6afn6akf0zqvo99rl33064ey7wgp39wz095
Content-Type: application/json
Content-Length: 67
{
   "stream-id":"2ondk9r49ke9uh3ur19ls1a8t3ywuw3uac6xai1xjawh21976s"
}
```

Figure 23. Get Server-Sent Events Stream Last Event ID: Request

```
200
Content-Type: application/json
Content-Length: 9
{
   "id":12
}
```

Figure 24. Get Server-Sent Events Stream Last Event ID: Response

Request aggregation services

Request aggregation services allow what would otherwise be multiple API requests to be submitted as a single request, with their multiple results likewise returned in a single response.

Submit Requests

The `Submit Requests` operation carries out multiple Web Services API requests as specified in its request body, and returns response information in a single API response.

HTTP method and URI

```
POST /api/services/aggregation/submit
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`requests`	Array of api-request objects	Required	Array of api-request objects, each representing one Web Services API operation for the service to run.
`req-headers`	Array of api-header objects	Optional	Array of api-header objects, each defining an HTTP header to add or modify in the header list for every api-request in the requests array. The headers associated with the `Submit Requests` operation are the basis for the headers used when carrying out each api-request. Any api-header objects specified here are processed in the order provided to modify those headers, and the resulting header set is used as the basis for every api-request. If a same-named header already appears in the base set of headers, the **value** supplied in the api-header object will override that existing value. If a same-named header does not already appear, then the header will be added with the specified value. An api-header object with only a **name** field and no **value** field will cause that header to be omitted. Note that an HTTP request header with an empty value can be set by supplying the empty string (**""**) for the api-header **value** field. Supplying **null** for the **value** field is not permitted. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC". The request header names **Content-Type** and **Accept** may not be specified. Only operations which expect JSON (if they have a request body), and generate JSON (if they have a response body) are supported. Therefore, the **Content-Type** and **Accept** headers associated with `Submit Requests` suffice. The request header name **Content-Length** may not be specified. The `Submit Requests` operation handles any needed calculation of **Content-Length** for the contained requests. Note that further header modification can be specified in each api-request. Such an individual request modification overrides any modification for same-named headers specified here.

Field name	Type	Rqd/Opt	Description
`resp-headers`	Array of String	Optional	HTTP header names. For each api-request, if the corresponding HTTP response contains any of these header names, those headers will be returned in the **headers** field of the corresponding api-response object in the `Submit Requests` response body. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC". Values may repeat in this array, and repeated values do not cause the **headers** field of the api-response objects to contain duplicates. However if a header name appears multiple times in an api-request's response, it will appear multiple times with each respective value in the **headers** field of the api-response. Note that, in general, response headers which are generated by the HMC API HTTP server itself, such as the **Server** response header, should not be expected to be seen for an individual api-response. That is because each individual api-request is not routed through the HTTP server but instead is driven as part of the `Submit Requests` HMC API HTTP server request. An exception to this guidance for server-generated response headers is the **Date** response header. If **Date** is requested in **resp-headers** it will be generated for each api-request by `Submit Requests` itself so that it may be returned.
`threads`	Integer (1-10)	Optional	The maximum number of threads the HMC may use to run the operations in the **requests** array. The default value is 1. A **threads** value of 1 specifies single-threaded execution. All requests will be issued on the HMC on a single thread in the same order they are specified in the **requests** array. A request will not be started until the previous request has completed. Single-threaded execution guarantees that the order of api-response objects in the response will match the order of the corresponding api-request objects in the **requests** array. A **threads** value greater than 1 specifies multi-threaded execution. The requests will be carried out on the HMC on multiple threads, up to the number specified. There will be no guarantee of the sequence of execution and no guarantee that any given request will be complete before another request is begun. The order of api-response objects in the response body will likewise not be predictable.

The api-request nested object contains the following fields:

Field name	Type	Rqd/Opt	Description
`method`	String	Required	The request method of the desired operation. For example, "GET", "POST", or "DELETE" (case is not significant).
`uri`	String/ URI	Required	The URI of the desired operation, including query parameters, if any.
`body`	Object	Optional	The JSON request body, if any, of the desired operation.
`req-headers`	Array of api-header objects	Optional	Array of api-header objects, each defining an HTTP header to add or modify in the header list used for this api-request only. The **req-headers** specified generally in the `Submit Requests` request body, if any, are applied first in the manner described for that field. Then these request-specific **req-headers** are applied to that modified base set in the same manner for this api-request only. The same guidance and restrictions given for the general **req-headers** field apply to this request-specific **req-headers** field.
`resp-headers`	Array of String	Optional	HTTP header names. For this api-request, if the corresponding HTTP response contains any of these header names, that header will be returned in the **headers** field of the corresponding api-response object in the `Submit Requests` response body. This array supplements any array specified generally in the `Submit Requests` request body.
`id`	String (1-64)	Optional	If an ID is provided, it will be returned in the **id** field of the corresponding api-response object in the `Submit Requests` response. Clients may use this ID to correlate api-request and api-response objects.

The api-header nested object contains the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	An HTTP header name. Note that when comparing header names case is not taken into account. So, for example, header name "abc" is considered the same as header name "ABC".
`value`	String	Optional	An HTTP header value.

The largest request body accepted by this operation is 256KB. Requests with bodies that exceed this maximum are rejected with an HTTP status 413 (Request Entity Too Large) response.

Response body contents

On successful completion, the response body is a JSON array of api-response objects as described below. The response body is sent using HTTP chunked transfer encoding.

If **threads** is not specified in the request body, or is specified with a value of 1, then the order of the api-response objects in the response corresponds to the order of the api-request objects in the **requests** array of the request body. So the first api-response object represents the response to the first api-request object, and so on.

If **threads** is specified in the request body with a value greater than 1, then no order of the api-response objects is guaranteed.

Regardless of the value of **threads**, the optional **id** field in the api-request and api-response objects may be used by the client to correlate requests and responses.

Each api-response object contains the following fields:

Field name	Type	Description
`body`	Object	The response body of the corresponding api-request. This field is always included but if there is no response body for the corresponding api-request then the value is **null**.
`status`	Integer	The HTTP status code for the corresponding api-request.
`headers`	Array of api-header objects	HTTP response headers for the corresponding api-request. This array contains only headers requested using the **resp-headers** field (global or specific to the api-request) in the `Submit Requests` request body. If a requested response header did not actually appear in the api-request's set of response headers, then it is omitted from this array. If no response headers fit this criterion then this field is omitted.
`id`	String	The **id** of the corresponding api-request object. If no **id** is provided in the api-request, this field is omitted.

Description

The `Submit Requests` request body specifies the Web Services API operations that are to be run in aggregation as an array of api-request objects. An api-request specification includes the operation **method** (for example, "GET", "POST", or "DELETE"), the operation URI, and the operation request body (if any). It may also include HTTP request header overrides in a **req-headers** field (in the event that those provided with `Submit Requests` itself are not satisfactory) and HTTP response header names in a **resp-headers** field (in the event that the client needs to know the value of a response header for an individual operation). Note that while you may specify these HTTP header behaviors for an individual operation in an api-request, you may also alternatively or additionally specify these HTTP header behaviors for every operation being aggregated. So, for example, if the client needs to receive the value of the **Location** response header for every operation, the **resp-headers** field does not need to be specified repeatedly in each api-request; the **resp-headers** field may be specified once as a peer to the **requests** array and it will apply to every api-request.

The `Submit Requests` response is an array of JSON objects, one for each API request specified in the request body. These JSON objects include the individual request's response body (if any) and status code. Optionally, they also include HTTP response headers associated with the individual request and an id that can be used to correlate that JSON object with the matching submitted request.

Note that only operations which expect JSON (if they have a request body), and generate JSON (if they have a response body) are supported for aggregation by `Submit Requests`. So operations that have request bodies that are not JSON (such as `Mount ISO Image`) and operations that have response bodies that are not JSON (such as `Get Metrics`) are not supported.

Single-threaded execution may be specified with a value of 1 for the **threads** field in the request body, or by omitting the **threads** field from the request body. Under single-threaded execution, `Submit`

`Requests` processes the requests sequentially in the order provided. It ensures each request has completed before beginning the next request (although some requests may initiate asynchronous jobs if they are so-specified). In other words, all of the requests are processed by the HMC on a single thread. Therefore the JSON objects in the response are in the same order as the requests in the request body.

Multi-threaded execution may be specified with a value greater than 1 for the **threads** field in the request body. Under multi-threaded execution, `Submit Requests` may process the requests in any order and their processing may overlap on up to the number of threads specified. Therefore, multi-threaded execution should not be specified if the sequence of execution of the requests is significant, or if there is some other reason why two or more of the requests should not be run simultaneously. Since the response order is not guaranteed, the client must use the **id** fields of api-request and api-response, or some other field unique to each expected response, to correlate the requests and responses.

Under both single-threaded and multi-threaded execution, requests are processed without regard to the result of any other request specified in the `Submit Requests` request body. Therefore, exercise caution aggregating requests with inter-dependencies. Even under single-threaded execution where it is guaranteed that request N has completed before request N+1 is initiated, it is *not* guaranteed that request N has completed *successfully* before request N+1 is initiated.

The response is returned using HTTP chunked transfer encoding, which allows the HMC to stream the response to each request as it completes, rather than having to accumulate the responses to all aggregated requests before returning data.

Authorization requirements

This operation has no authorization requirements of its own and will not directly report any authorization error.

The operation in each api-request is subject to that operation's authorization requirements as if it had been issued individually from the same API session. Any authorization problem is reflected in the status and/or body fields of the corresponding api-response object with the same content that would have been in the HTTP status code and response body had the request been issued directly and individually.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 147.

It is important to recognize that a successful status code from `Submit Requests` does not indicate that every (or, indeed, any) submitted request was successful. Each api-response object must be examined to determine the outcome of each submitted api-request.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 62. Submit Requests: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The request body attempted to use a **req-headers** field to modify one of the following request headers :**Content-Type**, **Accept**, or **Content-Length**.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/services/aggregation/submit HTTP/1.1
x-api-Session:  59ott72ftvzxkj7fz7g4jl48gh6csq7a7q3l98lm6gicq8wrl5
content-type: application/json
content-length: 257
{
   "requests":
    [
        {"id": "100000",
         "uri": "/api/console/users?name=stevef",
         "method": "get"},
        {"method": "DELETE",
         "uri": "/api/services/metrics/context/no-such-animal",
         "id": "200000"}
    ],
   "threads": 1,
   "req-headers": [{"name": "new-hdr", "value": "new-value"}]
}
```

Figure 25. Submit Requests: Request

```
200 OK
transfer-encoding: chunked
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 06 Apr 2018 15:08:04 GMT
content-type: application/json
[
    {
        "body":{"users":[{"name":"stevef","object-uri":"/api/users/a8ce7ac6-39ab-
                11e8-b4d7-00106f0d5d80","type":"standard"}]},
        "status":200,
        "id":"100000"
    },
    {
        "body":{"http-status":404,"reason":1,"request-uri":"/api/services/metrics/
                context/no-such-animal","request-method":"DELETE","message":"Context
                specified is not found, id is not a number","request-headers":{"new-hdr"
                :"new-value","content-length":"0","host":"127.0.0.1:6794","content-type"
                :"application/json","x-api-session":
                "59ott72ftvzxkj7fz7g4jl48gh6csq7a7q3l98lm6gicq8wrl5",
                "accept-encoding":"identity","accept":"application/json"},"request-body"
                :{},"request-authenticated-as":"acsadmin"},
        "status":404,
        "id":"200000"
    }
]
```

Figure 26. Submit Requests: Response

Usage notes

Note that specifying a higher number for **threads** in the request body will not necessarily lead to better performance. One reason is that a larger number of threads may increase the contention for access to the connection which delivers completed results to the client. The relative performance of different numbers of threads is largely dependent on attributes of the individual requests, making it impossible to recommend an ideal number of threads for all applications.

For example, it has been observed that for some large collections of quick-running requests, a thread count of 2 consistently provided better overall performance than either higher thread counts or single-threaded operation. Longer running requests that use different resources may see different results. Other processing load will of course influence performance as well. This variation is why `Submit Requests` allows the client to tune the maximum number of threads used.

Note further that increasing the number of threads for `Submit Requests` may increase the possibility of one or more of the constituent requests reporting a 503 (Service Unavailable) error with reason code

3. This error is reported when exceeding the number of concurrent API requests that are allowed, either for a given user or overall for an HMC. On its own, a single `Submit Requests` request will not exceed these limits, even if the maximum value of **threads** is specified. However, if there is other concurrent API activity, from the same client or other clients, then recognize that each thread running the constituent requests for `Submit Requests` counts individually toward the concurrent API request limits.

In summary, if multi-threaded behavior is desired, then the client application should specify the smallest number for **threads** that is observed to provide the desired performance most of the time. If in doubt, or if performance is not critical, defaulting to the single-threaded behavior is recommended.

Asynchronous job processing

Some of the operations that are provided in the Web Services API may take a significant amount of elapsed time to complete. In order to optimize the usage of HMC session resources and to allow the client application the opportunity to perform other processing, such long-running operations are structured to be executed asynchronously (rather than synchronously) from the perspective of the client application.

In a synchronous operation, the Web Services API does not respond to the client application's request until all of the processing associated with the request is complete (successfully or in error) and the API can provide a final result status for the operation. The client application thread is typically blocked (not running) during this time.

By contrast, in a function that operates asynchronously by starting a job, the Web Services API performs just the minimal front-end validation and set up work needed to accept the request to perform the indicated operation, and then quickly returns an HTTP 202 (Accepted) result to the client indicating that the operation request has been started but is not yet finished. Along with the HTTP 202 (Accepted) result, the client application is provided with a URI that represents the asynchronous job that is in progress. This URI is of the form `/api/jobs/{job-id}`.

At any point after receiving the HTTP 202 (Accepted) result, the client application can invoke the `Query Job Status` operation described in this section to determine if the job has ended or not. A job is considered ended if it runs to completion or is canceled. If the job has not yet ended, the `Query Job Status` request returns an indication that the job is still running or cancellation has been requested. If the job has ended, the `Query Job Status` request returns an indication of how the job ended along with the final status code, reason code and result data associated with the now-finished asynchronous processing. Once a job ends, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely.

Since the major reason an API operation is structured to be asynchronous is that it will take significant time to complete, very frequent polling for completion through calls to `Query Job Status` can lead to significant unproductive use of client application and HMC resources. In order to eliminate the need to poll at all, the Web Services API also provides asynchronous notifications of job completion or cancellation through its JMS and SSE notification capabilities. IBM recommends that client applications use this notification facility to determine when a job has ended rather than polling. See "Job completion notification" on page 84 for details on the JMS notification mechanism, or "Job completion notification" on page 94 for SSE.

If it is not practical for a client application to use asynchronous notification of job completion, the application should introduce elapsed-time delays between successive `Query Job Status` requests to poll the current job status in order to reduce unproductive use of resources.

Query Job Status

The `Query Job Status` operation returns the status associated with an asynchronous job. This operation is supported using the BCPii interface and is the only mechanism to obtain information for asynchronous jobs.

HTTP method and URI

```
GET /api/jobs/{job-id}
```

In this request, the URI variable *{job-id}* is the identifier of an asynchronous job associated with the API user, as returned in the response of the operation that initiated the job.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`status`	String Enum	An indication of the current disposition of the job. The possible values are as follows: • **"running"** - indicates that the job was found and it has not ended at the time of the query. • **"cancel-pending"** - indicates that the job was found and it has not ended but cancellation has been requested. • **"canceled"** - indicates that the job's normal course of execution was interrupted by a cancel request. The successful or unsuccessful completion of the job is indicated by the **job-status-code** and **job-reason-code** fields. • **"complete"** - indicates that the job was found and has completed running. The successful or error completion of the job is indicated by the **job-status-code** and **job-reason-code** fields.
`job-status-code`	Integer; Field provided only if status is **"complete"** or **"canceled"**	The job completion status code. This field is provided only if the **status** field is set to **"complete"** or **"canceled"**. This field provides the major status code describing the success or failure completion of the asynchronous action represented by the job. It is expressed in terms of an HTTP status code (i.e. the HTTP status code that would have been returned for the operation had it been performed synchronously). The values provided here and their meaning depend on the particular action that is being performed asynchronously. The description of these values is provided as part of the description for the operation that initiated the asynchronous job.
`job-reason-code`	Integer; Field provided only if status is **"complete"** or **"canceled"**	The job completion reason code. This field is provided only if the **status** field is set to **"complete"** or **"canceled"** and only if the **job-status-code** field indicates an error completion (status code other than 2XX). When present, this field provides a more detailed reason code describing the success or failure completion of the asynchronous action represented by the job. It is expressed in terms of the API reason code as are returned in standard error response bodies provided by the API. The values provided here and their meaning depend on the particular action that is being performed asynchronously. The description of these values is provided as part of the description or the operation that initiated the asynchronous job.

Field name	Type	Description
`job-results`	Object; Field provided only if status is **"complete"** or **"canceled"**	A nested object that provides results for the job. This field is provided only if the **status** field is set to **"complete"** or **"canceled"** and the asynchronous operation is documented to provide job results. If the **status** field is set to some other value, or the asynchronous operation provides no result information (beyond the job status and reason codes) then this field is omitted. The structure of the nested object provided by this field and its meaning depends on the particular action that is being performed asynchronously. The description of this object's structure is provided as part of the description of the operation that initiated the asynchronous job.

Description

The `Query Job Status` operation returns the status associated with an asynchronous job.

If the job designated by the URI is still running, the operation sets the **status** field in the response body to **"running"** and provides no other information about the job. If cancellation has been requested for the job designated by the URI but the cancellation action has not yet caused the job to end, the operation sets the **status** field in the response body to **"cancel-pending"** and provides no other information about the job. The client application may repeat the query at a later time, but should avoid frequent polling since that can lead to unproductive use of client and HMC resources. In order to eliminate the need to poll at all, the client application can (and should) use the asynchronous notifications facility provided by the API to receive notification that the job has ended through a JMS-based message or SSE-based event. See "Job completion notification" on page 84 for details on the JMS notification mechanism , or "Job completion notification" on page 94 for SSE.

If the job is complete, the operation sets the **status** field in the response body to **"complete"** and provides the other completion-related fields defined in the response body contents section above to report the results to the client application. If the job's normal execution sequence was interrupted by a cancel action, the operation sets the status field in the response body to **"canceled"** and provides the other related fields defined in the "Response body contents" on page 152 to report the results to the client application. Once a job ends, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely.

If the URI does not designate a job associated with the API user, HTTP status code 404 (Not Found) is returned to the client.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user or for the BCPii interface the requesting partition.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 152.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	2	For a job that was initiated using the BCPii interface the request does not contain or contains an invalid X-API-Target-Name header value.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/jobs/86e44546-107f-11e1-bde0-0010184c8334 HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4rymqa8ktzjep7dbyrll0k
```

Figure 27. Query Job Status: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Wed, 16 Nov 2011 18:19:35 GMT
content-type: application/json;charset=UTF-8
content-length: 63
{
   "job-reason-code": 0,
   "job-status-code": 200,
   "status": "complete"
}
```

Figure 28. Query Job Status: Response

Delete Completed Job Status

The `Delete Completed Job Status` operation deletes the job status and results associated with a job that has ended. This operation is supported using the BCPii interface.

HTTP method and URI

```
DELETE /api/jobs/{job-id}
```

In this request, the URI variable *{job-id}* is the identifier of an asynchronous job associated with the API user, as provided by the operation that initiated the job.

Description

The `Delete Completed Job Status` operation deletes the job status and results associated with a job that has ended.

If the job designated by the request URI has completed or has been canceled, its ending status and results are deleted from the HMC and status code 204 (No Content) is returned to the client.

If the job has not yet ended (i.e. is still running, or cancellation has been requested but is still pending), the operation fails and HTTP status code 409 (Conflict) is returned to the client.

If the URI does not designate a job associated with the API user, or if the job's status has already been deleted (either explicitly, or due to expiration of the status retention interval), HTTP status code 404 (Not Found) is returned to the client.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user or for the BCPii interface the requesting partition.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	2	For a job that was initiated using the BCPii interface the request does not contain or contains an invalid X-API-Target-Name header value.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.
409 (Conflict)	40	The URI designates an asynchronous job that has not ended.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Usage notes

- This operation is defined to operate only on jobs that have ended, i.e. have a **status** field with value **"complete"** or **"canceled"**. As a result, this operation cannot be used to cancel an in-progress asynchronous operation. See “Cancel Job” on page 156 for information pertaining to job cancellation.
- Once an asynchronous job has ended, job status is retained by the HMC for a minimum of 4 hours to allow the client application time to retrieve the results, but this status and results are not held indefinitely. At the expiration of the retention interval, job status is deleted as if the `Delete Completed Job Status` operation were called.

Example HTTP interaction

```
DELETE /api/jobs/86e44546-107f-11e1-bde0-0010184c8334 HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4rymqa8ktzjep7dbyrll0k
```

Figure 29. Delete Completed Job Status: Request

```
204 No Content
date: Wed, 16 Nov 2011 18:19:35 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 30. Delete Completed Job Status: Response

Cancel Job

The `Cancel Job` operation attempts to cancel an asynchronous job. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/jobs/{job-id}/operations/cancel
```

In this request, the URI variable *{job-id}* is the identifier of an asynchronous job associated with the API user, as provided by the operation that initiated the job.

Description

The `Cancel Job` operation attempts to cancel the specified job. The specific nature of the asynchronous job and its current state of execution when the request is received can affect the success of the cancellation action.

Not all asynchronous jobs support job cancellation. If a particular type of job supports cancellation, the description of the operation that initiates that type of job will explicitly specify that cancellation is supported and may describe other cancellation characteristics as well. If the description of the operation that initiates that type of job does not specify that cancellation is supported, then cancellation of that type of job is not possible.

If the specified job exists, but is of a type that does not support cancellation, status code 404 (Not Found) is returned and the job is allowed to continue processing without interruption.

If the specified job exists and has not yet completed (that is, the value of its **status** property is **"running"**), the cancellation request is made pending for the job, the **status** of the job is changed to **"cancel-pending"**, and HTTP status code 202 (Accepted) is returned. The processing of the pending cancellation request occurs asynchronously to the completion of the `Cancel Job` operation.

If the specified job exists and supports cancellation but either already has a cancellation request pending or has already ended (that is, has a **status** property with values **"cancel-pending"**, **"complete"** or **"canceled"**), HTTP status code 409 (Conflict) is returned with a reason code that more specifically indicates the particular error condition.

Once a cancellation request is made pending, the HMC will take steps to interrupt the processing of the job in order to cause the processing to end as quickly as is possible considering the nature of the processing done by the job. The conditions under which a running job can be interrupted vary depending on the type of job. For some types of jobs the console may be able to interrupt the processing very quickly while for others the console may be able to do so only as processing crosses selected interruption points. Thus, acceptance of a cancellation request does not guarantee that the processing of the job will either be immediately or eventually interrupted. As a consequence, it is possible that a job may proceed to normal completion (end with a **status** of **"complete"**) even after a cancellation request was accepted for the job.

Authorization requirements

This operation has the following authorization requirement:

- The job URI must designate an asynchronous job associated with the API user or for the BCPii interface the requesting partition.

HTTP status and reason codes

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	2	For a job that was initiated using the BCPii interface the request does not contain or contains an invalid X-API-Target-Name header value.
404 (Not Found)	1	The URI does not designate an asynchronous job associated with the API user.
	4	The URI designated an asynchronous job that can not be canceled.
409 (Conflict)	41	The URI designates an asynchronous job that has ended. It can not be canceled.
	42	The URI designates an asynchronous job that already has a cancellation request pending. A second cancel request is not allowed.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/jobs/86e44546-107f-11e1-bde0-0010184c8334/operations/cancel HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4rymqa8ktzjep7dbyrll0k
```

Figure 31. Cancel Job: Request

```
202 Accepted
date: Wed, 10 June 2015 18:19:35 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache
<No response body>
```

Figure 32. Cancel Job: Response

Chapter 8. Inventory and metrics services

The functions described in this chapter are termed "services" because unlike the interfaces described in many of the other chapters of this document, the functions described here are service-oriented rather than object-oriented in nature. That is, the functions of these services operate across multiple instances of managed objects rather than being directed at particular managed object instances.

The Inventory Service provides an efficient mechanism for retrieving identify and configuration information about all of the manageable resource instances that are managed by zManager. It provides this information in bulk form through a single request, and thus is expected to be a much more efficient means of determining this information than walking the entire resource tree one object at a time. It is anticipated that this service supports the requirements of a "discovery" phase of a client application that is interested in configuration information about all resources managed by zManager.

The Metrics Service provides a mechanism to retrieve performance metric data for resources that are managed by zManager. This data is captured periodically and buffered on the HMC. The data may include snapshots of performance data at a moment in time, or accumulated performance data, or both, as appropriate. This service is designed to support client applications that provide monitoring function for zManager managed resources.

Inventory services operations summary

The following operation is provided by the Inventory service:

Table 63. Inventory service: operations summary

Operation name	HTTP method and URI path
“Get Inventory” on page 160	`POST /api/services/inventory`

Metrics service operations summary

The following operations are provided by the Metrics service:

Table 64. Metrics service: operations summary

Operation name	HTTP method and URI path
“Create Metrics Context” on page 169	`POST /api/services/metrics/context`
“Get Metrics” on page 172	`GET /api/services/metrics/context/{metrics-context-id}`
“Delete Metrics Context” on page 175	`DELETE /api/services/metrics/context/{metrics-context-id}`

Table 65. Metrics service: URI variables

Variable	Description
{metrics-context-id}	Identifier of a metrics context object. Metrics contexts are associated with API sessions. Thus, this identifier is assigned by the metrics service so that it is unique within an API session and has a lifetime scoped to that session.

© Copyright IBM Corp. 2022, 2024

Inventory service

The Inventory Service is an API which allows the client application to fetch a list of managed resources and their properties.

This service is intended to support clients that need to determine the inventory and properties of all of the managed resources known to the HMC (or at least a large portion of those resources). Retrieving this information in bulk form using this service is expected to be much more efficient than walking the resource tree one object at a time using the object-oriented operations of the Web Services API.

The ability to filter the results to only certain classes of resources is provided.

A response to an inventory request is a series of JSON objects returned using HTTP chunked transfer encoding. These objects will be in a format specified in the corresponding resource class's inventory service data sections.

Resources returned are those to which the API client has object-level authorization.

Starting with API version 4.1 , limited API support is available for Adapters attached to a CPC that is not enabled for DPM. However, the **adapter** class does not provide adapters attached to a CPC that is not enabled for DPM.

Get Inventory

The `Get Inventory` operation fetches managed resources and associated properties.

HTTP method and URI

```
POST /api/services/inventory
```

Request body contents

The request body can include a specification of the classes of resources that should be returned. It contains the following field:

Field name	Type	Description
`resources`	Array of String Enum	An array of String values. Each element specifies a category or class of resource that should be returned. A category is simply a grouping of classes, so specifying a category is functionally equivalent to specifying all of its member classes. The request may include both categories and classes. Omitting the resources field, or providing an empty array, is equivalent to specifying an array listing all of the supported classes. Categories and associated class values: • Category: **"dpm-resources"** – Class: **"adapter"** – Class: **"partition"** – Class: **"partition-link"** – Class: **"virtual-switch"** – Class: **"storage-site"** – Class: **"storage-fabric"** – Class: **"storage-switch"** – Class: **"storage-subsystem"** – Class: **"storage-control-unit"** – Class: **"storage-group"** – Class: **"storage-template"** – Class: **"tape-link"** – Class: **"tape-library"** • Category: **"core-resources"** – Class: **"cpc"** – Class: **"logical-partition"** • Category: **"console-resources"** – Class: **"console"** – Class: **"custom-group"** – Class: **"user"** – Class: **"user-role"** • Category: **"certificate-resources"** [Added by feature **secure-boot-with-certificates**] – Class: **"secure-boot-certificate"**

Response body contents

On successful completion, the response body is a JSON array of JSON objects sent using HTTP chunked transfer encoding. The order in which these objects are returned is unspecified.

The array element documents are of 2 types:

- For resources that were successfully inventoried, the document will be as specified in the corresponding resource's inventory service data.
- For resources that were found but not successfully fully inventoried (i.e. the Object URI can be determined but not the properties), an inventory error document will be returned. Note that, even if

one or more of these inventory error documents is contained in the response, an HTTP status code of 200 (OK) is still returned. The fields in the inventory error document are:

Field name	Type	Description
`uri`	String/ URI	Canonical URI of the resource which could not be fully inventoried.
`class`	String	The class for these error documents is **"inventory-error"**.
`inventory-error-code`	Integer	A reason code for the inventory failure. Note that all of these reasons indicate success in locating a resource, but some sort of failure in gathering its properties during inventory collection. A subsequent call to get the properties for the URI in this error document may succeed. • 1: Resource not found on target. Although the resource's URI was located on the HMC, its properties were subsequently not located on the HMC or SE on which the property data for the managed object is to be gathered. • 2: Communication problem. Communication problems were experienced with the SE on which the property data for the managed object is to be gathered. • 3: Plugin load error. The code responsible for capturing the properties of a resource class experienced an unexpected problem loading. • 4: Unknown plugin error. The code responsible for capturing the properties of a resource returned an unrecognized error. • 5: Unexpected plugin error. The code responsible for capturing the properties of a resource returned an unexpected error. • 6: Timeout error. The code responsible for capturing the properties of a resource did not respond within the time allocated to it.
`inventory-error-text`	String	An error description for the inventory failure.
`inventory-error-details`	inventory-error-info Object	A nested inventory-error-info object that provides additional diagnostic information for unexpected inventory plugin errors. This field is provided if the **inventory-error-code** field is 5 (indicating unexpected plugin error). It is not provided for other **inventory-error-code** values. The format of the inventory-error-info object is defined in the next table.

The inventory-error-info object contains the following fields:

Field name	Type	Description
http-status	Integer	HTTP status code for the request.
request-uri	String	The URI that caused this error response.
reason	Integer	Numeric reason code providing more details as to the nature of the error) than is provided by the HTTP status code itself. This reason code is treated as a sub-code of the HTTP status code and thus must be used in conjunction with the HTTP status code to determine the error condition. Standard reason codes that apply across the entire API are described in"Common request validation reason codes" on page 66. Additional operation-specific reason codes may also be documented in the description of the specific API operations.

Field name	Type	Description
message	String	Message describing the error. This message is not currently localized.
stack	String	Internal HMC diagnostic information for the error. This field is supplied only on selected 5xx HTTP status codes.
error-details	Object	A nested object that provides additional operation-specific error information. This field is provided by selected operations, and the format of the nested object is as described by that operation.

Description

The Get Inventory operation returns information on managed resources and associated properties.

A resource is included in the response if it matches any one of the list of resource classes in the request body. Specifying a category is equivalent to specifying its member classes. If a class is repeated on the request, either explicitly or effectively through categories, the operation will behave as if the class were only specified once. If no resources are specified in the request body, all resources are returned. The inclusion of a resource may cause objects of certain related classes to also be included in the response. See the resource's Inventory Service Data section for the information about which, if any, related classes will be included.

Furthermore, a resource is included in the response only if the API user has object-access permission for that resource. If an HMC is a manager of a resource but the API user does not have permission to it, that resource is simply omitted from the response. A success status code is still returned.

If the HMC does not manage any resources to which the user has access, or if no resources are found that match the request body specification, an empty response is returned with a 204 (No Content) status code.

In addition to objects for inventoried resources, the response may include objects for resources whose URIs could be determined, but whose properties could not, for some reason, be obtained. Rather than treat these resources as completely non-inventoried and omit them, the URI and an error reason are returned.

The order in which the objects are returned is unspecified.

The Get Inventory implementation may choose to limit the number of simultaneous in-process inventory requests. If such a limit is reached, further requests will return an HTTP 503 (Service Unavailable) error status code until previous requests complete and the number of in-process inventory requests falls back below the limit.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to any object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 161. If there are no resources to provide, HTTP status code 204 (No Content) is returned, along with an empty response body.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
503 (Service Unavailable)	200	The request could not be processed because of the number of currently pending inventory requests. The request can be reissued at a later time.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Usage notes

The `Get Inventory` results represent a snapshot of inventory results as viewed from the HMC. The actual inventory can change, even as the results are being streamed back to the API client. Therefore, if the client wishes to stay informed about changes to the inventory and not risk missing any inventory changes, it should use the API event mechanisms to subscribe to inventory-related events before even issuing a `Get Inventory` request.

The `Get Inventory` results do not reflect all properties at a single moment in time. During the overall inventory collection process multiple resource's states and other properties may change. Therefore, states (or other properties) among two or more resources that might normally be expected to match (or have some other expected relationship) at one moment in time may instead return apparently inconsistent results in the `Get Inventory` response.

Example HTTP interaction

The following example illustrates a typical response for a `Get Inventory` request for the **logical-partition** class of resources. Responses for other classes will differ significantly from this because the data differs on a class by class basis. The format of the data returned by the Inventory Service for each class of object is described in a section entitled "Inventory service data" within the documentation for that object class.

```
POST /api/services/inventory HTTP/1.1
x-api-session: 2hatu4672fai3jicdxpeamt173qpqkkxglg8vyz0wfmnu2fx32
content-type: application/json
content-length: 36
{
   "resources":[
      "logical-partition"
   ]
}
```

Figure 33. Get Inventory: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 21:51:05 GMT
content-type: application/json;charset=UTF-8
[
   {
      "absolute-aap-capping":{
         "type":"none"
      },
      "absolute-cf-capping":{
         "type":"none"
      },
      "absolute-ifl-capping":{
         "type":"none"
      },
      "absolute-processing-capping":{
         "type":"none"
      },
      "absolute-ziip-capping":{
         "type":"none"
      },
      "acceptable-status":[
         "operating"
      ],
      "activation-mode":"not-set",
      "additional-status":"",
      "class":"logical-partition",
      "cluster-name":null,
      "current-aap-processing-weight":null,
      "current-aap-processing-weight-capped":null,
      "current-cf-processing-weight":null,
      "current-cf-processing-weight-capped":null,
      "current-ifl-processing-weight":null,
      "current-ifl-processing-weight-capped":null,
      "current-processing-weight":null,
      "current-processing-weight-capped":null,
      "current-ziip-processing-weight":null,
      "current-ziip-processing-weight-capped":null,
      "defined-capacity":null,
      "description":"LPAR Image",
      "group-profile-capacity":null,
      "group-profile-uri":null,
      "has-operating-system-messages":null,
      "has-unacceptable-status":true,
      "initial-aap-processing-weight":null,
      "initial-aap-processing-weight-capped":null,
      "initial-cf-processing-weight":null,
      "initial-cf-processing-weight-capped":null,
      "initial-ifl-processing-weight":null,
      "initial-ifl-processing-weight-capped":null,
      "initial-processing-weight":null,
      "initial-processing-weight-capped":null,
      "initial-ziip-processing-weight":null,
      "initial-ziip-processing-weight-capped":null,
      "is-locked":false,
      "last-used-activation-profile":"",
      "last-used-boot-record-logical-block-address":"C8",
      "last-used-disk-partition-id":0,
      "last-used-load-address":"00007",
      "last-used-load-parameter":"",
      "last-used-logical-unit-number":"1000000000000",
      "last-used-operating-system-specific-load-parameters":"SYSG",
      "last-used-world-wide-port-name":"50017380EB0B0141",
      "maximum-aap-processing-weight":null,
      "maximum-cf-processing-weight":null,
      "maximum-ifl-processing-weight":null,
      "maximum-processing-weight":null,
      "maximum-ziip-processing-weight":null,
      "minimum-aap-processing-weight":null,
```

Figure 34. Get Inventory: Response (Part 1)

```
      "minimum-cf-processing-weight":null,
      "minimum-ifl-processing-weight":null,
      "minimum-processing-weight":null,
      "minimum-ziip-processing-weight":null,
      "name":"VM137",
      "next-activation-profile-name":"VM137",
      "object-id":"d6641179-b8e0-3980-9d22-32cdd967c774",
      "object-uri":"/api/logical-partitions/d6641179-b8e0-3980-9d22-32cdd967c774",
      "os-ipl-token":null,
      "os-level":null,
      "os-name":null,
      "os-type":null,
      "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
      "partition-identifier":null,
      "partition-number":null,
      "program-status-word-information":null,
      "status":"not-activated",
      "storage-central-allocation":null,
      "storage-expanded-allocation":null,
      "sysplex-name":null,
      "workload-manager-enabled":null
   },
   {
      "absolute-aap-capping":{
         "type":"none"
      },
      "absolute-cf-capping":{
         "type":"none"
      },
      "absolute-ifl-capping":{
         "type":"none"
      },
      "absolute-processing-capping":{
         "type":"none"
      },
      "absolute-ziip-capping":{
         "type":"none"
      },
      "acceptable-status":[
         "operating"
      ],
      "activation-mode":"not-set",
      "additional-status":"",
      "class":"logical-partition",
      "cluster-name":null,
      "current-aap-processing-weight":null,
      "current-aap-processing-weight-capped":null,
      "current-cf-processing-weight":null,
      "current-cf-processing-weight-capped":null,
      "current-ifl-processing-weight":null,
      "current-ifl-processing-weight-capped":null,
      "current-processing-weight":null,
      "current-processing-weight-capped":null,
      "current-ziip-processing-weight":null,
      "current-ziip-processing-weight-capped":null,
      "defined-capacity":null,
      "description":"LPAR Image",
      "group-profile-capacity":null,
      "group-profile-uri":null,
      "has-operating-system-messages":null,
      "has-unacceptable-status":true,
```


Figure 35. Get Inventory: Response (Part 2)

```
        "initial-aap-processing-weight":null,
        "initial-aap-processing-weight-capped":null,
        "initial-cf-processing-weight":null,
        "initial-cf-processing-weight-capped":null,
        "initial-ifl-processing-weight":null,
        "initial-ifl-processing-weight-capped":null,
        "initial-processing-weight":null,
        "initial-processing-weight-capped":null,
        "initial-ziip-processing-weight":null,
        "initial-ziip-processing-weight-capped":null,
        "is-locked":false,
        "last-used-activation-profile":"",
        "last-used-boot-record-logical-block-address":"0",
        "last-used-disk-partition-id":0,
        "last-used-load-address":"00000",
        "last-used-load-parameter":"",
        "last-used-logical-unit-number":"0",
        "last-used-operating-system-specific-load-parameters":"",
        "last-used-world-wide-port-name":"0",
        "maximum-aap-processing-weight":null,
        "maximum-cf-processing-weight":null,
        "maximum-ifl-processing-weight":null,
        "maximum-processing-weight":null,
        "maximum-ziip-processing-weight":null,
        "minimum-aap-processing-weight":null,
        "minimum-cf-processing-weight":null,
        "minimum-ifl-processing-weight":null,
        "minimum-processing-weight":null,
        "minimum-ziip-processing-weight":null,
        "name":"MOBILEAU",
        "next-activation-profile-name":"MOBILEAU",
        "object-id":"3fd82f1a-ef7e-3e41-9a74-a5ce63089561",
        "object-uri":"/api/logical-partitions/3fd82f1a-ef7e-3e41-9a74-a5ce63089561",
        "os-ipl-token":null,
        "os-level":null,
        "os-name":null,
        "os-type":null,
        "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
        "partition-identifier":null,
        "partition-number":null,
        "program-status-word-information":null,
        "status":"not-activated",
        "storage-central-allocation":null,
        "storage-expanded-allocation":null,
        "sysplex-name":null,
        "workload-manager-enabled":null
    },
    {
        "absolute-aap-capping":{
            "type":"none"
        },
        "absolute-cf-capping":{
            "type":"none"
        },
        "absolute-ifl-capping":{
            "type":"none"
        },
        "absolute-processing-capping":{
            "type":"none"
        },
        "absolute-ziip-capping":{
            "type":"none"
        },
        "acceptable-status":[
            "operating"
        ],
```


Figure 36. Get Inventory: Response (Part 3)

```
        "activation-mode":"not-set",
        "additional-status":"",
        "class":"logical-partition",
        "cluster-name":null,
        "current-aap-processing-weight":null,
        "current-aap-processing-weight-capped":null,
        "current-cf-processing-weight":null,
        "current-cf-processing-weight-capped":null,
        "current-ifl-processing-weight":null,
        "current-ifl-processing-weight-capped":null,
        "current-processing-weight":null,
        "current-processing-weight-capped":null,
        "current-ziip-processing-weight":null,
        "current-ziip-processing-weight-capped":null,
        "defined-capacity":null,
        "description":"LPAR Image",
        "group-profile-capacity":null,
        "group-profile-uri":null,
        "has-operating-system-messages":null,
        "has-unacceptable-status":true,
        "initial-aap-processing-weight":null,
        "initial-aap-processing-weight-capped":null,
        "initial-cf-processing-weight":null,
        "initial-cf-processing-weight-capped":null,
        "initial-ifl-processing-weight":null,
        "initial-ifl-processing-weight-capped":null,
        "initial-processing-weight":null,
        "initial-processing-weight-capped":null,
        "initial-ziip-processing-weight":null,
        "initial-ziip-processing-weight-capped":null,
        "is-locked":false,
        "last-used-activation-profile":"",
        "last-used-boot-record-logical-block-address":"0",
        "last-used-disk-partition-id":0,
        "last-used-load-address":"00000",
        "last-used-load-parameter":"",
        "last-used-logical-unit-number":"0",
        "last-used-operating-system-specific-load-parameters":"",
        "last-used-world-wide-port-name":"0",
        "maximum-aap-processing-weight":null,
        "maximum-cf-processing-weight":null,
        "maximum-ifl-processing-weight":null,
        "maximum-processing-weight":null,
        "maximum-ziip-processing-weight":null,
        "minimum-aap-processing-weight":null,
        "minimum-cf-processing-weight":null,
        "minimum-ifl-processing-weight":null,
        "minimum-processing-weight":null,
        "minimum-ziip-processing-weight":null,
        "name":"T257",
        "next-activation-profile-name":"T257",
        "object-id":"130e0c22-97ec-3a37-a109-93e1e2a4f5df",
        "object-uri":"/api/logical-partitions/130e0c22-97ec-3a37-a109-93e1e2a4f5df",
        "os-ipl-token":null,
        "os-level":null,
        "os-name":null,
        "os-type":null,
        "parent":"/api/cpcs/38f77df5-ecea-3063-89bb-d0865dc4c881",
        "partition-identifier":null,
        "partition-number":null,
        "program-status-word-information":null,
        "status":"not-activated",
        "storage-central-allocation":null,
        "storage-expanded-allocation":null,
        "sysplex-name":null,
        "workload-manager-enabled":null
    }
]
```


Figure 37. Get Inventory: Response (Part 4)

Metrics service

The zEnterprise® (or later) Central Processing Complexes (CPCs) and their associated system resources are instrumented at key points to collect performance and utilization data. The data is forwarded by the metric data providers to a buffer on the HMC where it is made available to consumers of this API.

The data collection instrumentation organizes and formalizes the data it collects into a series of named metric groups. The Metrics Service API allows specification of the metric groups the client wishes to query. The API returns some information about the format of the metrics that are being fetched. Specifically, a structure called a metrics context is associated with any metrics retrieval, and that structure includes metric group names, individual metric field names, and the associated individual metric data types.

The full metric group documentation, however, including descriptions of the data collected and the frequency of collection, can be found in Chapter 9, “Metric groups,” on page 177.

Create Metrics Context

The `Create Metrics Context` operation creates a context under which metrics can be repeatedly retrieved. This context will be associated with the API session under which it was created.

HTTP method and URI

```
POST /api/services/metrics/context
```

Request body contents

A request body must be specified. It has the following fields:

Field name	Type	Description
`anticipated-frequency-seconds`	Integer	The number of seconds the client anticipates will elapse between `Get Metrics` calls against this context. The minimum accepted value is 15.
`metric-groups`	Array of Strings	Optional. Array of metric group names. If specified, then results from future `Get Metrics` requests associated with this context will be limited to only metrics with group names matching one of the specified values. If not specified, or if an empty array is specified, then results will not be limited with respect to metric group names.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`metrics-context-uri`	String/ URI	Canonical URI path of the metrics context object created by this operation This includes the metrics-context-id. E.g. "/api/services/metrics/context/1", where "1" is the metrics-context-id.
`metric-group-infos`	Array of objects	Array of metric-group-info objects (described in the next table) that describe the data format for each metric group that may be returned by future GETs associated with this metric context.

Each nested metric-group-info object contains the following fields:

Field name	Type	Description
`group-name`	String	The name of the metric group for which we are providing descriptive information.
`metric-infos`	Array	Array of metric-info objects (described in the next table). These describe each metric for the group in the order that they will appear in future GETs associated with this context.

Each nested metric-info object contains the following fields:

Field name	Type	Description
`metric-name`	String	The name of the metric.
`metric-type`	String Enum	One of the following, describing the type of the metric: **"boolean-metric"**, **"byte-metric"**, **"double-metric"**, **"long-metric"**, **"integer-metric"**, **"short-metric"**, **"string-metric"** See the Get Metrics "Response body contents" on page 172 for further information on these metric types.

Description

This operation establishes a context for future `Get Metrics` operations that is valid for the current API session. Because of the high frequency of invocation and large volume of data expected, the metrics service interface has been structured to optimize the transmission of data on each `Get Metrics` request. Thus, rather than use a self-describing representation for the results returned by each Get Metrics, the metrics service instead provides the descriptive metadata as results from this `Create Metrics Context` operation. It then returns the metric data in a compact format each time `Get Metrics` is invoked.

At a high level, the `Create Metrics Context` response communicates two primary pieces of information back to the client. One is the metrics-context-uri, which includes the ID of the metrics context that must be referenced on future GETs to associate them with this context. The other is the metric-groups description data. That data provides the metric type and metric name information for each metric group whose metrics may be returned by this context. This may be useful to the client for determining how to parse future `Get Metrics` responses for this context, although the full documentation on metric group formats is found in Chapter 9, "Metric groups," on page 177.

The anticipated-frequency-seconds specification which is required on the request body tells the metrics service how frequently the client anticipates issuing `Get Metrics` requests against this context. The metrics service may take no action based on this frequency, but reserves the right to invalidate and delete the metrics context if 4 times the specified frequency passes without receipt of an associated `Get Metrics` operation.

Optional result filtering is provided by field metric-groups on the request body. If a non-empty metric-groups arrays is specified, then future `Get Metrics` operations associated with this context will return only groups with names listed there.

Additionally, if a metric-groups array of group names is specified on the `Create Metrics Context` request, then the response JSON document will include only matching metric-group-info fields. If one or more names in the metric-groups array does not represent a metric group registered on the HMC, then HTTP error status code 400 (Bad Request) will be returned and the context will not be established.

Although the POST response fully describes and guarantees the ordering of metric-infos within a metric group for that context, as a matter of policy the HMC will further guarantee that, for a given metric group, any additions of new metrics to the group will be to the end of the list for the group.

Authorization requirements

There are no authorization restrictions on creating a metrics context. However any future metric results returned by `Get Metrics` queries against that context will be restricted to managed objects accessible according to the permissions associated with the API session under which the metrics context was established.

Note that there is no indication through an HTTP status or reason code that future results may be restricted due to authorization restrictions. Rather, success is indicated and future `Get Metrics` responses behave just as if any restricted objects did not exist.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 169. The URI for the newly created context is also provided in the **Location** header of the response.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/services/metrics/context HTTP/1.1
x-api-session: 3dajvxld29sk9zi584isbiguj4rl4gfkrsuxjprnsymb44e4vd
content-type: application/json
content-length: 73
{
   "anticipated-frequency-seconds":30,
   "metric-groups":[
      "channel-usage"
   ]
}
```

Figure 38. Create Metrics Context: Request

```
201 Created
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
location: /api/services/metrics/context/1
cache-control: no-cache
date: Wed, 18 Sep 2019 22:45:15 GMT
content-type: application/json;charset=UTF-8
{
   "metric-group-infos":[
      {
         "group-name":"channel-usage",
         "metric-infos":[
            {
               "metric-name":"channel-name",
               "metric-type":"string-metric"
            },
            {
               "metric-name":"shared-channel",
               "metric-type":"boolean-metric"
            },
            {
               "metric-name":"logical-partition-name",
               "metric-type":"string-metric"
            },
            {
               "metric-name":"channel-usage",
               "metric-type":"integer-metric"
            }
         ]
      }
   ],
   "metrics-context-uri":"/api/services/metrics/context/1"
}
```

Figure 39. Create Metrics Context: Response

Get Metrics

The `Get Metrics` operation retrieves the current set of metrics associated with an established metrics context.

HTTP method and URI

```
GET /api/services/metrics/context/{metrics-context-id}
```

In this request, the URI variable *{metrics-context-id}* is the identifier of the metrics context object for which metrics are to be obtained.

Response body contents

On successful completion, the response body contains the set of metrics associated with the metrics context. The response is sent using HTTP chunked transfer encoding and UTF-8 character encoding. A MIME media type of **application/vnd.ibm-z-zmanager-metrics** is used and is specified in the **Content-Type** header on the response.

Because performance and scalability are a major concern for the metrics service, the response body is in a terse custom format, rather than being presented as a JSON object. The data type, name, and order information required to parse and interpret the response is provided in a previous `Create Metrics Context` response.

Data in this format will be delimited by newlines and commas.

Using a partial Extended Backus-Naur Form, where a comma (,) indicates concatenation and curly braces ({}) indicate 0 or more repetitions, we can express the format this way:

```
MetricsResponse = {MetricsGroup},NL

MetricsGroup = MetricsGroupName,NL,{ObjectValues},NL
MetricsGroupName = StringValue

NL = “\n”

ObjectValues = ObjectURI,NL,Timestamp,NL,ValueRows,NL

Timestamp = LongValue

ObjectURI = StringValue

ValueRows = ValueRow,{ValueRow}
ValueRow = Value,{“,”,Value},NL

Value = BooleanValue | ByteValue | DoubleValue | LongValue | IntegerValue | ShortValue |
StringValue
```

The MetricsGroupName is the name of the metrics group, as a StringValue as defined below.

The Timestamp is the time when the associated values were buffered (i.e. "cached") on the HMC. It is expressed as an "epoch" timestamp: a LongValue giving the milliseconds since January 1, 1970, 00:00:00 GMT (just as is expected, for example, by the constructor of a java.util.Date object).

The ObjectURI is the canonical URI of the object, as a StringValue as defined below.

NL is a single newline character (Unicode U+000A).

All the varieties of `Value` will be represented as strings according to the following rules and limits:

- BooleanValue
  - Either the string true or the string false.
- ByteValue
  - A string representation of a signed decimal integer in the range -128 to 127 (i.e. the range of a signed 8 bit integer).
- DoubleValue
  - A string representation of a 64 bit IEEE 754 floating point number in the range +/-4.9E-324 to +/-3.4028235E+38. Note that, although IEEE 754 provides for representations of positive or negative Infinity and NaN, such values are not allowed in the metric data feed and thus will not appear in a metrics service result. For results with a magnitude greater than or equal to 10^-3 and less than 10^7, the string representation will be a dotted decimal (e.g. 1.7, -32.467). For results with magnitudes outside that range, the string representation will be computerized scientific notation (e.g. -4.23E127).
- LongValue
  - A string representation of a signed decimal integer in the range -9223372036854775808 to 9223372036854775807 (i.e. the range of a signed 64 bit integer).
- IntegerValue
  - A string representation of a signed decimal integer in the range -2147483648 to 2147483647 (i.e. the range of a signed 32 bit integer).
- ShortValue
  - A string representation of a signed decimal integer in the range -32768 to 32767 (i.e. the range of a signed 16 bit integer).
- StringValue
  - A string starting with a double-quote, ending with a double-quote, and with any embedded double-quotes or backslashes escaped with a preceding backslash (i.e. escaped as \" and \\).

Description

On successful execution status code 200 (OK) is returned, with a response body as described above.

The URI path on the request must designate an existing metrics context for the current API session. If the URI designates an unrecognized context for the API session, then status code 404 (Not Found) is returned.

Note that under some circumstances the metrics service may delete a metrics context, requiring the client to establish a new context in order to resume metric retrievals. For example, the metrics service may choose to delete a given context if the time since the last associated `Get Metrics` request has exceeded 4 times the anticipated frequency specified when the context was created. In addition, the client may explicitly delete a metrics context with the `Delete Metrics Context` operation. If the URI designates a context that was once valid for the current API session, but no longer is, then status code 409 (Conflict) is returned.

Authorization requirements

Only metrics referencing managed objects accessible according to the permissions associated with the API session under which the Get Metrics is being issued will be returned. Note that there is no indication through an HTTP status or reason code that results may have been restricted due to authorization restrictions. Rather, success is indicated and the responses are just as if any restricted objects did not exist.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 172.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The metrics context ID in the URI (*{metrics-context-id}*) does not designate a metrics context for the associated API session.
409 (Conflict)	1	The metrics context ID in the URI (*{metrics-context-id}*) designates a metrics context for the associated API session that is no longer valid.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage notes

- Repeated metrics retrievals, even consecutive retrievals against a common metrics context, will not necessarily yield metrics for the exact same set of objects. Objects can come and go from the system's inventory due to various circumstances unrelated to the metrics data. The metrics feed for a particular metric group may stop for some reason, and the metrics data may therefore expire from the HMC's buffer (i.e. the metrics cache). The access permissions of a user associated with a metrics context may change, giving the user access to a smaller or larger set of objects (and therefore, perhaps, a smaller or larger set of metrics data).
- It is possible that there may be no metrics to return for one or more metric groups associated with the specified metrics context. For example, data for a metric group may not be buffered on the HMC at the time of the Get Metrics invocation, or authorization restrictions related to objects in a requested

metric group may prevent any metrics for that group from being returned. If there is no metric data to be returned for a metric group name, then that group name does not appear in the response body. Furthermore, if there is no metric data to return for any metric group name associated with a context, the response body format above specifies that the body will consist only of a single newline character. This is nonetheless considered a successful response. In other words, an HTTP status code 200 (OK) will still be returned with such a response.

Example HTTP interaction

```
GET /api/services/metrics/context/1 HTTP/1.1
x-api-session: 4rtgkmzk62b6kz0351uhz5fytb8e0o9weagn4rpj4adv52pf0s
```

Figure 40. Get Metrics: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 22:46:52 GMT
content-type: application/vnd.ibm-z-zmanager-metrics;charset=UTF-8
"channel-usage"
"/api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5"
1568847192154
"0.00",true,"Shared",1
"0.01",true,"Shared",1
"0.04",true,"Shared",1
"0.05",true,"Shared",1
"0.08",true,"Shared",1
"0.09",true,"Shared",1
"0.0C",true,"Shared",1
"0.0D",true,"Shared",1
"0.10",true,"Shared",1
"0.11",true,"Shared",1
"0.14",true,"Shared",1
"0.15",true,"Shared",1
"0.18",true,"Shared",1
"0.19",true,"Shared",1
"0.1C",true,"Shared",1
"0.1D",true,"Shared",1
"0.24",true,"Shared",1
"1.25",false,"LP22    ",1
"0.28",true,"Shared",1
"0.29",true,"Shared",1
"0.2C",true,"Shared",1
"0.2D",true,"Shared",1
"0.30",true,"Shared",1
"0.31",true,"Shared",1
"0.34",true,"Shared",1
"0.35",true,"Shared",1
"0.E2",true,"Shared",10
"0.E3",true,"Shared",10
"0.FC",false,"CF01    ",1
"0.FD",false,"LP01    ",1
"0.FE",false,"CF02    ",1
"1.FF",false,"LP17    ",1
<3 blank lines here (consecutive new lines)>
```

Figure 41. Get Metrics: Response

Delete Metrics Context

The `Delete Metrics Context` operation deletes a metrics context.

HTTP method and URI

```
DELETE /api/services/metrics/context/{metrics-context-id}
```

In this request, the URI variable *{metrics-context-id}* is the identifier of the metrics context object for which metrics are to be obtained.

Description

This operation deletes the metrics context ID. That is, it disassociates it from the API session and cleans up any data associated with it. Further `Get Metrics` requests against this context will result in status code 409 (Conflict).

The URI path must designate an existing valid metrics context for the current API session. If the URI path represents an already invalidated metrics context for the current API session, status code 409 (Conflict) is returned. If the URI path does not represent a recognized metrics context for the current API session, status code 404 (Not Found) is returned.

Authorization requirements

There are no authorization requirements for deleting a metrics context. The association with the API session is implicit, so there is no possibility of deleting a context that was created by a different API session. In other words, only the session which created a metrics context can delete it.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The metrics context ID in the URI (*{metrics-context-id}*) does not designate a metrics context for the associated API session.
409 (Conflict)	1	The metrics context ID in the URI (*{metrics-context-id}*) designates a metrics context for the associated API session that is no longer valid.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/services/metrics/context/1 HTTP/1.1
x-api-session: 6a9oz3ymut6rvjijrft0loqhfzgpp0rnu4mjishwh6d39jh31q
```

Figure 42. Delete Metrics Context: Request

```
204 No Content
date: Wed, 07 Dec 2011 04:01:59 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 43. Delete Metrics Context: Response

Chapter 9. Metric groups

This chapter provides a description of the metric groups that can be retrieved using the Metrics Service. For each metric group provided by the HMC, the material in this chapter describes the purpose and general characteristics of the metric group, and then defines the content of the metric group through a table that specifies the metric fields provided by the group. The order in which metric fields appear within these tables corresponds to the order in which the data items appear in a value row returned by the `Get Metrics` operation. For example, the metric field appearing in the first row of a metric group table (and identified in a table below as being in position 1) will be the first data item provided in a value row for that metric group; the metric field appearing in the second row (position 2) will be the next data item in a value row, and so on. Thus, the order in which metric fields are documented here is considered semantically significant and can be relied upon by client applications in order to simplify parsing of the data retrieved using the `Get Metrics` operation.

The contents of metric groups may be extended in future versions of this API. If a metric group is extended, new metric fields will be added to the end so as to not alter the relative positions of any of the existing fields. Such new fields would not be understood by a client application designed for an earlier version of the API. Therefore, applications must be developed using the philosophy of "ignore what you don't understand/expect" when processing metric group data in order to tolerate such possible future extensions. See “Compatibility” on page 6 for more discussion on API compatibility principles.

Monitors dashboard metric groups

The Monitors Dashboard task is the current system monitoring interface on the HMC. It gives a dashboard display to monitor system resources, such as power consumption, environmental data, processor usage, etc.

In order to provide programmatic access to this same data, the utilization and environment data that is displayed on the user interface is also provided through the Metrics Service in the following metric groups.

Channels

This metric group reports the channel usage for each channel on the system. An instance of this metric group is created for each channel of a CPC.

Metric Group Name
"channel-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 66. Channels metric group

Pos	Metric field name	Type	Units	Description
1	**channel-name**	String	—	The name of the channel in the form CSS.Chpid
2	**shared-channel**	Boolean	—	True if the channel is shared among logical partitions, and false if it is not

© Copyright IBM Corp. 2022, 2024

Table 66. Channels metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
3	**logical-partition-name**	String	—	For channel types for which logical partition names are available, the name of the owning logical partition or the value **"shared"** if the channel is shared. For other channel types for which the name is not available (for example, coupling channels), the value is always an empty string.
4	**channel-usage**	Integer	%	The channel percent usage (0 – 100%)

CPC overview

This metric group reports the aggregated processor usage and channel usage, the ambient temperature, and total system power consumption for each system. The **cpc-processor-usage** is the average of the percentages of processing capacity for all the physical processors in the CPC. The **channel-usage** is the average of the percentages of I/O capacity for all the channels and adapters in the CPC.

Metric Group Name
"cpc-usage-overview"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 67. CPC overview metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**cpc-processor-usage**	Integer	%	The processor percent usage.
2	**channel-usage**	Integer	%	The channel percent usage.
3	**power-consumption-watts**	Integer	Watts	The total system power consumption.
4	**temperature-celsius**	Double	Degrees Celsius	The ambient temperature.
5	**cp-shared-processor-usage**	Integer	%	The processor percent usage for all CP shared processors. Set to -1 if there are no processors of this type.
6	**cp-dedicated-processor-usage**	Integer	%	The processor percent usage for all CP dedicated processors. Set to -1 if there are no processors of this type.
7	**ifl-shared-processor-usage**	Integer	%	The processor percent usage for all IFL shared processors. Set to -1 if there are no processors of this type.
8	**ifl-dedicated-processor-usage**	Integer	%	The processor percent usage for all IFL dedicated processors. Set to -1 if there are no processors of this type.
9	**icf-shared-processor-usage**	Integer	%	The processor percent usage for all ICF shared processors. Set to -1 if there are no processors of this type.

Table 67. CPC overview metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
10	**icf-dedicated-processor-usage**	Integer	%	The processor percent usage for all ICF dedicated processors. Set to -1 if there are no processors of this type.
11	**iip-shared-processor-usage**	Integer	%	The processor percent usage for all zIIP shared processors. Set to -1 if there are no processors of this type.
12	**iip-dedicated-processor-usage**	Integer	%	The processor percent usage for all zIIP dedicated processors. Set to -1 if there are no processors of this type.
13	**aap-shared-processor-usage**	Integer	%	The processor percent usage for all zAAP shared processors. Set to -1 if there are no processors of this type.
14	**aap-dedicated-processor-usage**	Integer	%	The processor percent usage for all zAAP dedicated processors. Set to -1 if there are no processors of this type.
15	**all-shared-processor-usage**	Integer	%	The processor percent usage for all shared processors. Set to -1 if there are no processors of this type.
16	**all-dedicated-processor-usage**	Integer	%	The processor percent usage for all dedicated processors. Set to -1 if there are no processors of this type.
17	**cp-all-processor-usage**	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
18	**ifl-all-processor-usage**	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
19	**icf-all-processor-usage**	Integer	%	The processor percent usage for all ICF processors. Set to -1 if there are no processors of this type.
20	**iip-all-processor-usage**	Integer	%	The processor percent usage for all zIIP processors. Set to -1 if there are no processors of this type.

DPM system overview

This metric group reports the aggregated processor usage, network usage, storage usage, accelerator usage, crypto usage, power consumption and temperature for each DPM enabled system.

Metric Group Name
"dpm-system-usage-overview"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 68. DPM system overview metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**processor-usage**	Integer	%	The processor percent usage.
2	**network-usage**	Integer	%	The network percent usage. Set to -1 if there are no adapters of this type.
3	**storage-usage**	Integer	%	The storage percent usage. Set to -1 if there are no adapters of this type.
4	**accelerator-usage**	Integer	%	The accelerator percent usage. Set to -1 if there are no adapters of this type.
5	**crypto-usage**	Integer	%	The crypto percent usage. Set to -1 if there are no adapters of this type.
6	**power-consumption-watts**	Integer	Watts	The power consumption in watts.
7	**temperature-celsius**	Double	Degrees Celsius	The ambient temperature.
8	**cp-shared-processor-usage**	Integer	%	The processor percent usage for all CP shared processors. Set to -1 if there are no processors of this type.
9	**cp-all-processor-usage**	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
10	**ifl-shared-processor-usage**	Integer	%	The processor percent usage for all IFL shared processors. Set to -1 if there are no processors of this type.
11	**ifl-all-processor-usage**	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
12	**all-shared-processor-usage**	Integer	%	The processor percent usage for all shared processors. Set to -1 if there are no processors of this type.

Logical partitions

This metric group reports the processor usage, z/VM paging rate for each active logical partition (aka Image, LPAR Image, Zone, PR/SM virtual server) on the system, and power consumption. [Updated by feature **enviromental-metrics**].

Metric Group Name
"logical-partition-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"logical-partition"

The following metrics are provided in each entry of this metric group:

Table 69. Logical partitions metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**processor-usage**	Integer	%	The processor percent usage.

Table 69. Logical partitions metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
2	**zvm-paging-rate**	Integer	Pages Per Second	The z/VM paging rate. Only returned for logical partitions running z/VM level 6.1 or higher that have the appropriate agent running in them. **Note:** As of API version 3.1, this metric is no longer available. It will remain in this metric group for compatibility reasons, and its value will always be -1.
3	**cp-processor-usage**	Integer	%	The processor percent usage for all CP processors. Set to -1 if there are no processors of this type.
4	**ifl-processor-usage**	Integer	%	The processor percent usage for all IFL processors. Set to -1 if there are no processors of this type.
5	**icf-processor-usage**	Integer	%	The processor percent usage for all ICF processors. Set to -1 if there are no processors of this type.
6	**iip-processor-usage**	Integer	%	The processor percent usage for all IIP processors. Set to -1 if there are no processors of this type.
9	**power-consumption**	Integer	Watts	The total power utilized by logical partitions. [Added by feature **environmental-metrics**]

Partitions

This metric group reports the processor usage, network usage, storage usage, accelerator usage, and crypto usage for each active partition on a DPM enabled system.

Metric Group Name
"partition-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"partition"

The following metrics are provided in each entry of this metric group:

Table 70. Partitions metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**processor-usage**	Integer	%	The processor percent usage.
2	**network-usage**	Integer	%	The network percent usage. Set to -1 if there are no adapters of this type.
3	**storage-usage**	Integer	%	The storage percent usage. Set to -1 if there are no adapters of this type.

Table 70. Partitions metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
4	**accelerator-usage**	Integer	%	The accelerator percent usage. Set to -1 if there are no adapters of this type.
5	**crypto-usage**	Integer	%	The crypto percent usage. Set to -1 if there are no adapters of this type.

zCPC environmentals and power

This metric group reports environmental data and power consumption for the zCPC.

Metric Group Name
"zcpc-environmentals-and-power"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 71. zCPC environmentals and power metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**temperature-celsius**	Double	Degrees Celsius	The ambient temperature
2	**humidity**	Integer	%	The relative humidity
3	**dew-point-celsius**	Double	Degrees Celsius	The dew point
4	**power-consumption-watts**	Integer	Watts	The power consumption in watts
5	**heat-load**	Integer	Btu/hour	The total heat load of the system (heat load forced-air + heat load water)
6	**heat-load-forced-air**	Integer	Btu/hour	The heat load covered by forced-air
7	**heat-load-water**	Integer	Btu/hour	The heat load covered by water
8	**exhaust-temperature-celsius**	Double	Degrees Celsius	The exhaust temperature
10	**total-partition-power-consumption**	Integer	Watts	The total power utilized by components assigned to individual partitions. [Added by feature **environmental-metrics**]
11	**total-infrastructure-power-consumption**	Integer	Watts	The total power consumption of infrastructure components (including top of rack switches, SE/HMAs, and PDUs) and should not be accounted to partition power. [Added by feature **environmental-metrics**]

Table 71. zCPC environmentals and power metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
12	**total-unassigned-power-consumption**	Integer	Watts	The total unassigned power consumption of unused I/O adapters and components that are not assigned to any partition (including standby components). [Added by feature **environmental-metrics**]

Power status

This metric group reports line cord power information of connected Power Distribution Units (PDU) or BPAs (Bulk Power Assembly) in the system.

Metric Group Name
"environmental-power-status"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 72. Power status metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**linecord-one-name**	String	None	Line cord One identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
2	**linecord-one-power-phase-A**	Integer	Watts	Power in Phase A for line cord One
3	**linecord-one-power-phase-B**	Integer	Watts	Power in Phase B for line cord One
4	**linecord-one-power-phase-C**	Integer	Watts	Power in Phase C for line cord One
5	**linecord-two-name**	String	None	Line cord Two identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
6	**linecord-two-power-phase-A**	Integer	Watts	Power in Phase A for line cord Two
7	**linecord-two-power-phase-B**	Integer	Watts	Power in Phase B for line cord Two
8	**linecord-two-power-phase-C**	Integer	Watts	Power in Phase C for line cord Two

Table 72. Power status metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
9	**linecord-three-name**	String	None	Line cord Three identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
10	**linecord-three-power-phase-A**	Integer	Watts	Power in Phase A for line cord Three
11	**linecord-three-power-phase-B**	Integer	Watts	Power in Phase B for line cord Three
12	**linecord-three-power-phase-C**	Integer	Watts	Power in Phase C for line cord Three
13	**linecord-four-name**	String	None	Line cord Four identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
14	**linecord-four-power-phase-A**	Integer	Watts	Power in Phase A for line cord Four
15	**linecord-four-power-phase-B**	Integer	Watts	Power in Phase B for line cord Four
16	**linecord-four-power-phase-C**	Integer	Watts	Power in Phase C for line cord Four
17	**linecord-five-name**	String	None	Line cord Five identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
18	**linecord-five-power-phase-A**	Integer	Watts	Power in Phase A for line cord Five
19	**linecord-five-power-phase-B**	Integer	Watts	Power in Phase B for line cord Five
20	**linecord-five-power-phase-C**	Integer	Watts	Power in Phase C for line cord Five
21	**linecord-six-name**	String	None	Line cord Six identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
22	**linecord-six-power-phase-A**	Integer	Watts	Power in Phase A for line cord Six

Table 72. Power status metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
23	**linecord-six-power-phase-B**	Integer	Watts	Power in Phase B for line cord Six
24	**linecord-six-power-phase-C**	Integer	Watts	Power in Phase C for line cord Six
25	**linecord-seven-name**	String	None	Line cord Seven identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
26	**linecord-seven-power-phase-A**	Integer	Watts	Power in Phase A for line cord Seven
27	**linecord-seven-power-phase-B**	Integer	Watts	Power in Phase B for line cord Seven
28	**linecord-seven-power-phase-C**	Integer	Watts	Power in Phase C for line cord Seven
29	**linecord-eight-name**	String	None	Line cord Eight identifier. Set to **"not-connected"** if line cord is not available. If line cord is not available power phases are set to 0.
30	**linecord-eight-power-phase-A**	Integer	Watts	Power in Phase A for line cord Eight
31	**linecord-eight-power-phase-B**	Integer	Watts	Power in Phase B for line cord Eight
32	**linecord-eight-power-phase-C**	Integer	Watts	Power in Phase C for line cord Eight

zCPC processors

This metric group reports the processor usage for each physical zCPC processor on the system. This includes the System Assist Processors (SAPs). An instance of this metric group is created for each processor of a CPC.

Metric Group Name
"zcpc-processor-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 73. zCPC processors metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**processor-name**	String		The name of the zCPC processor in the form processor-type + processor ID. For example, IFL01.
2	**processor-type**	String		The type of zCPC processor. The valid types are: **"cp"**, **"ifl"**, **"icf"**, **"iip"**, **"aap"**, and **"sap"**. The valid types for DPM enabled systems are: **"cp"**, **"ifl"**, and **"sap"**.
3	**processor-usage**	Integer	%	The processor percent usage.
4	**smt-usage**	Integer	%	The percentage of time the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.
5	**thread-0-usage**	Integer	%	The percent usage of thread 0 when the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.
6	**thread-1-usage**	Integer	%	The percent usage of thread 1 when the processor is running in simultaneous multithreading (SMT) mode. Set to -1 when SMT mode is not supported by the CPC.

Cryptos

This metric group reports the adapter usage for each crypto on the system. An instance of this metric group is created for each crypto adapter. This metric group is not used for a DPM system. For DPM, crypto adapters are reported in the Adapters metric group. See “Adapters” on page 186.

Metric Group Name
"crypto-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 74. Crypto metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**channel-id**	String		The physical channel identifier of the crypto
2	**crypto-id**	String		The crypto identifier of the crypto, decimal 0-15
3	**adapter-usage**	Integer	%	The adapter percent usage (0-100%)

Adapters

This metric group reports the adapter usage for each adapter on the DPM enabled system. An instance of this metric group is created for each adapter.

Metric Group Name
"adapter-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"adapter"

The following metrics are provided in each entry of this metric group:

Table 75. Adapters metric group

Pos	Metric field name	Type	Units	Description
1	**adapter-usage**	Integer	%	The adapter percent usage

Flash memory adapters

This metric group reports the adapter usage for each Flash memory (Flash Express) adapter on the system. An instance of this metric group is created for each Flash memory adapter of the CPC. If a CPC has no flash memory adapters, then no data will appear in this metric group for that CPC.

Note: Flash Express has a planned exploitation of December 2012.

Metric Group Name
"flash-memory-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 76. Flash memory adapters metric group

Pos	Metric field name	Type	Units	Description
1	**channel-id**	String		The physical channel identifier of the Flash memory adapter
2	**adapter-usage**	Integer	%	The adapter percent usage (0-100%)

RoCE adapters

This metric group reports the adapter usage for each RoCE (10GbE RoCE) adapter on the system. Metrics are collected from these adapters and provided to the user, only on systems where DPM is not enabled. An instance of this metric group is created for each RoCE adapter of the CPC.

Note: This metric group is not populated and will be removed in a future HMC version.

Metric Group Name
"roce-usage"

Collection Interval
15 seconds

Applicable Managed Object Class
"cpc"

The following metrics are provided in each entry of this metric group:

Table 77. RoCE adapters metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**channel-id**	String		The physical channel identifier of the RoCE adapter
2	**adapter-usage**	Integer	%	The adapter percent usage (0-100%)

Network management metrics

Following are the network management metric groups.

Network adapter port metric group

OSA and RoCE network adapters have up to two physical ports that connect to the network. Metrics are collected from these ports on a DPM enabled system and provided to the user. This metrics group will contain metrics data representing metrics for one physical port. Metrics are collected and provided on an interval, and each metric provided is the total cumulative value, and not a delta.

Metric Group Name
"network-physical-adapter-port"

Collection Interval
30 seconds

Applicable Managed Object Class
"adapter"

The following metrics are provided in each entry of this metric group:

Table 78. Network adapter port metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**
1	**network-port-id**	Integer		Numerical value corresponding to the network adapter's physical port. For OSA adapters, this value can be either 0 or 1, and for RoCE adapters, this value can be 1 or 2. To get more information about the physical port, a URI can be constructed using this value together with the target **adapter-id**: `/api/adapters/{adapter-id}/network-ports/{network-port-id}`
2	**bytes-sent**	Long	Bytes	Number of bytes this physical port sent out to the attached network.
3	**bytes-received**	Long	Bytes	Number of unicast packets this physical port received from the attached network.
4	**packets-sent**	Long	Count	Number of unicast packets this physical port sent out to the attached network.
5	**packets-received**	Long	Count	Number of unicast packets this physical port received from the attached network.

Table 78. Network adapter port metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
6	**packets-sent-dropped**	Long	Count	Number of packets that were dropped when this physical port was sending them out to the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
7	**packets-received-dropped**	Long	Count	Number of packets that were dropped when this physical port was receiving them from the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
8	**packets-sent-discarded**	Long	Count	Number of packets that were discarded when this physical port was sending them out to the attached network. Packets may be discarded due to errors such as malformed packets.
9	**packets-received-discarded**	Long	Count	Number of packets that were discarded when this physical port was receiving them from the attached network. Packets may be discarded due to errors such as malformed packets.
10	**multicast-packets-sent**	Long	Count	Number of multicast packets this physical port sent out to the attached network.
11	**multicast-packets-received**	Long	Count	Number of multicast packets this physical port received from the attached network.
12	**broadcast-packets-sent**	Long	Count	Number of broadcast packets this physical port sent out to the attached network.
13	**broadcast-packets-received**	Long	Count	Number of broadcast packets this physical port received from the attached network.
14	**interval-bytes-sent**	Long	Bytes	Number of bytes sent by this physical port over the collection interval.
15	**interval-bytes-received**	Long	Bytes	Number of bytes received by this physical port over the collection interval.
16	**bytes-per-second-sent**	Long	Bytes per second	Number of bytes sent per second by this physical port over the collection interval.
17	**bytes-per-second-received**	Long	Bytes per Second	Number of bytes per second received by this physical port over the collection interval.
18	**utilization**	Long	Percentage	Link utilization expressed as usage percentage of overall link bandwidth.
19	**mac-address**	String		The MAC address of this uplink, if known. If it is not known, then **"N/A"**.

Table 78. Network adapter port metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
20	**flags**	Long		Flags indicating the types of metrics that are supported by this interface. The value of this field should be interpreted as a bitmask. The meaning of each bit is as follows: • 0x02 - Byte counts are supported • 0x04 - Packet counts are supported • 0x08 - Drop counts are supported • 0x10 - Discard counts are supported • 0x20 - Multicast counts are supported • 0x40 - Broadcast counts are supported • 0x80 - Interval bytes sent and received are supported

Network interface metric group

This metric group reports metrics for NICs on a DPM enabled system. NICs are network resources associated with DPM partitions. Only NICs that are activated will report metric data. This metrics group will contain metrics data representing metrics for one NIC. Metrics are collected and provided on an interval, and each metric provided is the total cumulative value, and not a delta.

Metric Group Name
"partition-attached-network-interface"

Collection Interval
30 seconds

Applicable Managed Object Class
"nic"

The following metrics are provided in each entry of this metric group:

Table 79. Network interface metric group				
Pos	**Metric field name**	**Type**	**Units**	**Description**

1	**partition-id**	String (36)		The unique identifier for the partition that owns the NIC whose metric is contained within this metric group. To get information about the partition, the URI can be constructed using **partition-id**:
				`/api/partitions/{partition-id}`
				To get information about the NIC, the URI can be constructed using the **partition-id** value together with the target **nic-id**:
				`/api/partitions/{partition-id}/nics/{nic-id}`

2	**bytes-sent**	Long	Bytes	Number of bytes this network adapter sent out to the attached network.

Table 79. Network interface metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
3	**bytes-received**	Long	Bytes	Number of bytes this network adapter received from the attached network.
4	**packets-sent**	Long	Count	Number of unicast packets this network adapter sent out to the attached network.
5	**packets-received**	Long	Count	Number of unicast packets this network adapter received from the attached network.
6	**packets-sent-dropped**	Long	Count	Number of packets that were dropped when this network adapter was sending them out to the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
7	**packets-received-dropped**	Long	Count	Number of packets that were dropped when this network adapter was receiving them from the attached network. Packets may be dropped due to conditions related to resource constraints such as a buffer shortage.
8	**packets-sent-discarded**	Long	Count	Number of packets that were discarded when this network adapter was sending them out to the attached network. Packets may be discarded due to errors such as malformed packets.
9	**packets-received-discarded**	Long	Count	Number of packets that were discarded when this network adapter was receiving them from the attached network. Packets may be discarded due to errors such as malformed packets.
10	**multicast-packets-sent**	Long	Count	Number of multicast packets this network adapter sent out to the attached network.
11	**multicast-packets-received**	Long	Count	Number of multicast packets this network adapter received from the attached network.
12	**broadcast-packets-sent**	Long	Count	Number of broadcast packets this network adapter sent out to the attached network.
13	**broadcast-packets-received**	Long	Count	Number of broadcast packets this network adapter received from the attached network.
14	**interval-bytes-sent**	Long	Bytes	Number of bytes sent by this network adapter over the collection interval.
15	**interval-bytes-received**	Long	Bytes	Number of bytes received by this network adapter over the collection interval.
16	**bytes-per-second-sent**	Long	Bytes per second	Number of bytes sent per second by this network adapter over the collection interval.

Table 79. Network interface metric group (continued)				
Pos	**Metric field name**	**Type**	**Units**	**Description**
17	**bytes-per-second-received**	Long	Bytes per Second	Number of bytes per second received by this network adapter over the collection interval.
18	**flags**	Long		Flags indicating the types of metrics that are supported by this interface. The value of this field should be interpreted as a bitmask. The meaning of each bit is as follows: • 0x02 - Byte counts are supported • 0x04 - Packet counts are supported • 0x08 - Drop counts are supported • 0x10 - Discard counts are supported • 0x20 - Multicast counts are supported • 0x40 - Broadcast counts are supported • 0x80 - Interval bytes sent and received are supported

Part 3. CPC management

Topics in this part describe CPC management.

Topics covered in this part are:

- Chapter 10, “Dynamic Partition Manager (DPM),” on page 195
- Chapter 11, “Core IBM zSystems resources,” on page 781
- Chapter 12, “Energy management,” on page 1383

© Copyright IBM Corp. 2022, 2024

Chapter 10. Dynamic Partition Manager (DPM)

IBM Dynamic Partition Manager (DPM) expands on the PR/SM (Processor Resource/Systems Manager) concept of logical partitions and provides a simplified, consumable, enhanced user experience reducing the barriers of adoption for new and existing clients.

IBM DPM seamlessly integrates platform I/O resource management, server, network and storage resource provisioning, dynamic resource adjustments and resource monitoring. This enhanced, and encapsulated management through the HMC as the single management end-point eliminates the need for special skills to operate the integrated IBM zSystems platform and its hardware and virtual infrastructure. The HMC Web Services API enables customers, or IBM and third-party data center management tooling respectively to automate all of the IBM DPM tasks, for example, to discover, manage, and monitor resources, and to provide event notifications.

Read the chapters in Part 4 of the *Dynamic Partition Manager (DPM) Guide*, for prerequisites for enabling DPM on a mainframe system, and information about supported functions. This part also includes migration instructions and information about I/O adapter configuration.

The DPM APIs provide access to and control of the following HMC/SE objects:

- Partition object
- Virtual Function element object
- NIC element object
- HBA element object
- Adapter object
- Network Port element object
- Storage Port element object
- Virtual Switch object
- Capacity Group element object
- FICON Storage Configuration objects:

  Storage Site object
  Storage Fabric object
  Storage Switch object
  Storage Subsystem object
  Storage Control Unit object
- Storage Group object
- Storage Template object
- Tape Library object
- Tape Link object

The following Adapter operations are also available for adapters whose parent CPC is not enabled for DPM:

- `List Permitted Adapters`
- `Update Adapter Firmware`

FICON storage configuration

A number of the classes defined in the Dynamic Partition Manager chapter are interrelated and in their entirety, define the FICON storage configuration as seen by a single CPC. This configuration is used by the CPC to understand the paths that connect their storage adapters to the storage resources that are needed by the partitions running in that CPC.

© Copyright IBM Corp. 2022, 2024

The following diagram illustrates the major components of a FICON configuration.

- FICON storage resources are organized into storage sites. The primary site is the one in which the CPC (System "M01" in the diagram) is located. It likely also includes sets of storage switches and storage subsystems that are also local to the CPC. The primary site will always exist. Optionally, a second alternate site can also exist. Alternate sites are typically remote to the CPC, and often used for redundancy. In the diagram, the primary site is labeled "New York" and the alternate site is labeled "New Jersey".
- Storage subsystems represent the physical storage units. They are physically connected (cabled) to a set of storage switches or directly to a set of CPC storage adapters. Storage subsystems are subdivided into logical control units, which provide access to a subset of a subsystem's storage resources. Storage control units are logically connected to CPC storage adapters, optionally through one or two storage switches. In the diagram, storage subsystems named "DS8870 A" and "DS8886 A" are located in the primary "New York" site and storage subsystems named "DS8870 B" and "DS8886 B" are located in the alternate "New Jersey" site.
- Storage switches are optionally used within a CPC storage adapter to storage control unit connection. The switch that is connected to the CPC storage adapter is referred to as the "entrance" switch. The switch that is connected to the storage subsystem is referred to as the "exit" switch. The entrance and exit switches can be the same, which means there is exactly one switch connecting the adapter and control unit. If the entrance and exit switches are different, there are two switches connecting the adapter and control unit, and they are referred to as being in a cascaded switch configuration. In a cascaded configuration, the entrance switch will be in the primary site and the exit switch will be in the alternate site. A storage switch exists in exactly one storage site and in exactly one storage fabric. In the diagram, storage switches 10 and 20 exist in the primary "New York" site. The CPC is connected through those switches to the "DS8870 A" and "DS8886 A" storage subsystems. Switches 11 and 21 exist in the alternate "New Jersey" site. The CPC is connected in cascaded configurations through those switches to the "DS8870 B" and "DS8886 B" storage subsystems.
- The set of all storage switches that are interconnected defines a storage fabric. In a multi-site configuration, switches from both sites are interconnected, therefore in such configurations, a fabric will span multiple sites. In the diagram, storage switches 10 and 11 are interconnected and define storage fabric "Fabric A". Storage switches 20 and 21 define storage fabric "Fabric B".

It is important to understand that a FICON configuration is scoped to a single CPC, and represents that CPC's view of a set of storage resources. A second CPC's FICON configuration may include objects that represent the same physical or logical entities, but they will be returned to an API client as separate objects. The data models for those objects provide no intrinsic way to determine those objects represent the same physical or logical entity. Correlation is possible only if the configurations are created using names or other identifiers that are the same.

Device number constraints

Many of the DPM objects and elements objects have **device-number** properties that allow an API client to view and/or modify the device number for the devices represented by those objects. In such cases, the values of the **device-number** properties for all devices associated with a single partition are always constrained. There are two distinct device number namespaces for channel-based and PCI-based devices.

Channel-based device numbers

The device numbers for all channel-based devices associated with a single partition must be unique. The following objects and element objects are channel-based:

- All Partition NIC elements of **type "iqd"** or **"osd"**.
- All Partition HBA elements.
- All Storage Group Storage Volume elements of storage groups of **type "fc"**.
- All Storage Group Virtual Storage Resource elements.
- All Tape Link Virtual Tape Resource elements.

PCI-based device numbers

The device numbers for all PCI-based devices associated with a single partition must be unique. The following objects and element objects are PCI-based:

- All Partition NIC elements of **type "roce"** or **"cna"**.
- All Partition Virtual Function elements.
- All Storage Group Storage Volume elements of storage groups of **type "nvme"**.

Operations summary

Following are the operations summaries for each of the Dynamic Partition Manager (DPM) objects.

Partition operations summary

The following table provides an overview of the operations provided for Partition objects.

Table 80. DPM - Partition: operations summary

Operation name	HTTP method and URI path
“List Partitions of a CPC” on page 234	`GET /api/cpcs/{cpc-id}/partitions`
“List Permitted Partitions” on page 237	`GET /api/console/operations/list-permitted-partitions`
“Create Partition” on page 239	`POST /api/cpcs/{cpc-id}/partitions`
“Delete Partition” on page 245	`DELETE /api/partitions/{partition-id}`
“Delete Partition Asynchronously” on page 247	`POST /api/partitions/{partition-id}/operations/async-delete`
“Get Partition Properties” on page 250	`GET /api/partitions/{partition-id}`

Table 80. DPM - Partition: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Update Partition Properties" on page 253	POST /api/partitions/*{partition-id}*
"Update Partition Properties Asynchronously" on page 257	POST /api/partitions/*{partition-id}*/operations/async-update
"Start Partition" on page 262	POST /api/partitions/*{partition-id}*/operations/start
"Stop Partition" on page 269	POST /api/partitions/*{partition-id}*/operations/stop
"Dump Partition" on page 271	POST /api/partitions/*{partition-id}*/operations/scsi-dump
"Start Dump Program" on page 275	POST /api/partitions/*{partition-id}*/operations/start-dump-program
"Perform PSW Restart" on page 281	POST /api/partitions/*{partition-id}*/operations/psw-restart
"Create Virtual Function" on page 283	POST /api/partitions/*{partition-id}*/virtual-functions
"Delete Virtual Function" on page 286	DELETE /api/partitions/*{partition-id}*/virtual-functions/*{virtual-function-id}*
"Get Virtual Function Properties" on page 287	GET /api/partitions/*{partition-id}*/virtual-functions/*{virtual-function-id}*
"Update Virtual Function Properties" on page 289	POST /api/partitions/*{partition-id}*/virtual-functions/*{virtual-function-id}*
"Create NIC" on page 291	POST /api/partitions/*{partition-id}*/nics
"Delete NIC" on page 296	DELETE /api/partitions/*{partition-id}*/nics/*{nic-id}*
"Get NIC Properties" on page 298	GET /api/partitions/*{partition-id}*/nics/*{nic-id}*
"Update NIC Properties" on page 300	POST /api/partitions/*{partition-id}*/nics/*{nic-id}*
"Increase Crypto Configuration" on page 305	POST /api/partitions/*{partition-id}*/operations/increase-crypto-configuration
"Change Crypto Domain Configuration" on page 308	POST /api/partitions/*{partition-id}*/operations/change-crypto-domain-configuration
"Decrease Crypto Configuration" on page 310	POST /api/partitions/*{partition-id}*/operations/decrease-crypto-configuration
"Zeroize Crypto Domain" on page 313	POST /api/partitions/*{partition-id}*/operations/zeroize-crypto-domain
"Mount ISO Image" on page 316	POST /api/partitions/*{partition-id}*/operations/mount-iso-image
"Unmount ISO Image" on page 317	POST /api/partitions/*{partition-id}*/operations/unmount-iso-image
"Attach Storage Group to Partition" on page 266	POST /api/partitions/*{partition-id}*/operations/attach-storage-group

Table 80. DPM - Partition: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Detach Storage Group from Partition" on page 319	`POST /api/partitions/{partition-id}/operations/detach-storage-group`
"Create HBA" on page 321	`POST /api/partitions/{partition-id}/hbas`
"Delete HBA" on page 324	`DELETE /api/partitions/{partition-id}/hbas{hba-id}`
"Update HBA Properties" on page 326	`POST /api/partitions/{partition-id}/hbas/{hba-id}`
"Get HBA Properties" on page 328	`GET /api/partitions/{partition-id}/hbas/{hba-id}`
"Reassign Storage Adapter Port" on page 330	`POST /api/partitions/{partition-id}/hbas/{hba-id}/operations/reassign-storage-adapter-port`
"Send OS Command" on page 332	`POST /api/partitions/{partition-id}/operations/send-os-cmd`
"Open OS Message Channel" on page 334	`POST /api/partitions/{partition-id}/operations/open-os-message-channel`
"List OS Messages of a Partition" on page 336	`GET /api/partitions/{partition-id}/operations/list-os-messages`
"Delete Partition OS Message" on page 339	`POST /api/partitions/{partition-id}/operations/delete-os-message`
"Get ASCII Console WebSocket URI" on page 341	`POST /api/partitions/{partition-id}/operations/get-ascii-console-websocket-uri`
"Attach Tape Link to Partition" on page 343	`POST /api/partitions/{partition-id}/operations/attach-tape-link`
"Detach Tape Link from Partition" on page 346	`POST /api/partitions/{partition-id}/operations/detach-tape-link`
"Report a Partition Problem" on page 348	`POST /api/partitions/{partition-id}/operations/report-problem` [Added by feature **report-a-problem**]
"Get Partition Historical Sustainability Data" on page 350	`POST /api/partitions/{partition-id}/operations/get-historical-sustainability-data` [Added by feature **environmental-metrics**]
"Assign Certificate to Partition" on page 353	`POST /api/partitions/{partition-id}/operations/assign-certificate` [Added by feature **secure-boot-with-certificates**]
"Unassign Certificate from Partition" on page 355	`POST /api/partitions/{partition-id}/operations/unassign-certificate` [Added by feature **secure-boot-with-certificates**]

Table 81. DPM - Partition: URI variables	
Variable	**Description**
{cpc-id}	Object ID of the CPC object.

Table 81. DPM - Partition: URI variables (continued)	
Variable	**Description**
{partition-id}	Object ID of the Partition object.
{virtual-function-id}	Element ID of the Virtual Function element object.
{nic-id}	Element ID of the NIC element object.
{hba-id}	Element ID of the HBA element object.

Adapter operations summary

The following table provides an overview of the operations provided for Adapter objects.

Table 82. DPM - Adapter: operations summary	
Operation name	**HTTP method and URI path**
"List Adapters of a CPC" on page 374	`GET /api/cpcs/{cpc-id}/adapters`
"List Permitted Adapters" on page 377	`GET /api/console/operations/list-permitted-adapters`
"Get Adapter Properties" on page 381	`GET /api/adapters/{adapter-id}`
"Update Adapter Properties" on page 383	`POST /api/adapters/{adapter-id}}`
"Change Crypto Type" on page 386	`POST /api/adapters/{adapter-id}/operations/change-crypto-type`
"Create Hipersocket" on page 388	`POST /api/cpcs/{cpc-id}/adapters`
"Delete Hipersocket" on page 390	`DELETE /api/adapters/{adapter-id}`
"Get Partitions Assigned to Adapter" on page 392	`GET /api/adapters/{adapter-id}/operations/get-partitions-assigned-to-adapter`
"Get Network Port Properties" on page 394	`GET /api/adapters/{adapter-id}/network-ports/{network-port-id}`
"Update Network Port Properties" on page 396	`POST /api/adapters/{adapter-id}/network-ports/{network-port-id}`
"Get Storage Port Properties" on page 398	`GET /api/adapters/{adapter-id}/storage-ports/{storage-port-id}`
"Update Storage Port Properties" on page 399	`POST /api/adapters/{adapter-id}/storage-ports/{storage-port-id}`
"Change Adapter Type" on page 401	`POST /api/adapters/{adapter-id}/operations/change-adapter-type`
"Update Adapter Firmware" on page 403	`POST /api/adapters/{adapter-id}/operations/update-firmware`

Table 83. DPM - Adapter: URI variables	
Variable	**Description**
{cpc-id}	Object ID of a CPC object.
{adapter-id}	Object ID of an Adapter object.
{network-port-id}	Element ID of a Network Port element object.
{storage-port-id}	Element ID of a Storage Port element object.

Virtual Switch operations summary

The following table provides an overview of the operations provided for Virtual Switch objects.

Table 84. DPM - Virtual Switch: operations summary	
Operation name	**HTTP method and URI path**
"List Virtual Switches of a CPC" on page 409	`GET /api/cpcs/{cpc-id}/virtual-switches`
"List Permitted Virtual Switches" on page 411	`GET /api/console/operations/list-permitted-virtual-switches` [Added by feature **dpm-hipersockets-partition-link-management**]
"Get Virtual Switch Properties" on page 414	`GET /api/virtual-switches/{vswitch-id}`
"Get Connected VNICs of a Virtual Switch" on page 416	`GET /api/virtual-switches/{vswitch-id}/operations/get-connected-vnics`
"Update Virtual Switch Properties" on page 417	`POST /api/virtual-switches/{vswitch-id}`

Table 85. DPM - Virtual Switch: URI variables	
Variable	**Description**
{cpc-id}	Object ID of the CPC object
{vswitch-id}	Object ID of the Virtual Switch object

Capacity Group operations summary

The following table provides an overview of the operations provided for Capacity Group objects.

Table 86. DPM - Capacity Group: operations summary	
Operation name	**HTTP method and URI path**
"List Capacity Groups of a CPC" on page 421	`GET /api/cpcs/{cpc-id}/capacity-groups`
"Create Capacity Group" on page 423	`POST /api/cpcs/{cpc-id}/capacity-groups`
"Delete Capacity Group" on page 426	`DELETE /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}`
"Get Capacity Group Properties" on page 427	`GET /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}`

Table 86. DPM - Capacity Group: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Add Partition to Capacity Group" on page 429	`POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/add-partition`
"Remove Partition from Capacity Group" on page 431	`POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/remove-partition`
"Update Capacity Group Properties" on page 433	`POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}`

Table 87. DPM - Capacity Group: URI variables	
Variable	**Description**
{cpc-id}	Object ID of a CPC object
{capacity-group-id}	Element ID of the Capacity Group element object

Storage Site operations summary

The following table provides an overview of the operations provided for Storage Site objects.

Table 88. DPM - Storage Site: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Sites" on page 437	`GET /api/storage-sites`
"Create Storage Site" on page 439	`POST /api/storage-sites`
"Delete Storage Site" on page 442	`DELETE /api/storage-sites`
"Get Storage Site Properties" on page 444	`GET /api/storage-sites/{storage-site-id}`
"Update Storage Site Properties" on page 445	`POST /api/storage-sites/{storage-site-id}`

Table 89. DPM - Storage Site: URI variables	
Variable	**Description**
{storage-site-id}	Object ID of a Storage Site object

Storage Fabric operations summary

The following table provides an overview of the operations provided for Storage Fabric objects.

Table 90. DPM - Storage Fabric: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Fabrics" on page 449	`GET /api/storage-fabrics`
"Create Storage Fabric" on page 451	`POST /api/storage-fabrics`
"Delete Storage Fabric" on page 454	`DELETE /api/storage-fabrics/{storage-fabric-id}`
"Get Storage Fabric Properties" on page 455	`GET /api/storage-fabrics/{storage-fabric-id}`

Table 90. DPM - Storage Fabric: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Update Storage Fabric Properties" on page 457	`POST /api/storage-fabrics/{storage-fabric-id}`

Table 91. DPM - Storage Fabric: URI variables	
Variable	**Description**
{storage-fabric-id}	Object ID of a Storage Fabric object

Storage Switch operations summary

The following table provides an overview of the operations provided for Storage Switch objects.

Table 92. DPM - Storage Switch: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Switches of a Storage Site" on page 461	`GET /api/storage-sites/{storage-site-id}/storage-switches`
"List Storage Switches of a Storage Fabric" on page 463	`GET /api/storage-fabrics/{storage-fabric-id}/storage-switches`
"Define Storage Switch" on page 466	`POST /api/console/operations/define-storage-switch`
"Undefine Storage Switch" on page 468	`POST /api/storage-switches/{storage-switch-id}/operations/undefine`
"Get Storage Switch Properties" on page 470	`GET /api/storage-switches/{storage-switch-id}`
"Update Storage Switch Properties" on page 471	`POST /api/storage-switches/{storage-switch-id}`
"Move Storage Switch to Storage Site" on page 473	`POST /api/storage-switches/{storage-switch-id}/operations/move-storage-site`
"Move Storage Switch to Storage Fabric" on page 475	`POST /api/storage-switches/{storage-switch-id}/operations/move-storage-fabric`

Table 93. DPM - Storage Switch: URI variables	
Variable	**Description**
{storage-site-id}	Object ID of a Storage Site object
{storage-fabric-id}	Object ID of a Storage Fabric object
{storage-switch-id}	Object ID of a Storage Switch object

Storage Subsystem operations summary

The following table provides an overview of the operations provided for Storage Subsystem objects.

Table 94. DPM - Storage Subsystem: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Subsystems of a Storage Site" on page 480	`GET /api/storage-sites/{storage-site-id}/storage-subsystems`
"Define Storage Subsystem" on page 482	`POST /api/console/operations/define-storage-subsystem`
"Undefine Storage Subsystem" on page 484	`POST /api/storage-subsystems/{storage-subsystem-id}/operations/undefine`
"Get Storage Subsystem Properties" on page 486	`GET /api/storage-subsystems/{storage-subsystem-id}`
"Update Storage Subsystem Properties" on page 488	`POST /api/storage-subsystems/{storage-subsystem-id}`
"Move Storage Subsystem to Storage Site" on page 489	`POST /api/storage-subsystems/{storage-subsystem-id}/operations/move-storage-site`
"Add Connection Endpoint" on page 491	`POST /api/storage-subsystems/{storage-subsystem-id}/operations/add-connection-endpoint`
"Remove Connection Endpoint" on page 494	`POST /api/storage-subsystems/{storage-subsystem-id}/operations/remove-connection-endpoint`

Table 95. DPM - Storage Subsystem: URI variables	
Variable	**Description**
{storage-site-id}	Object ID of a Storage Site object
{storage-subsystem-id}	Object ID of a Storage Subsystem object

Storage Control Unit operations summary

The following table provides an overview of the operations provided for Storage Control Unit objects.

Table 96. DPM - Storage Control Unit: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Control Units of a Storage Subsystem" on page 500	`GET /api/storage-subsystems/{storage-subsystem-id}/storage-control-units`
"Define Storage Control Unit" on page 502	`POST /api/storage-subsystems/{storage-subsystem-id}/operations/define-storage-control-unit`
"Undefine Storage Control Unit" on page 504	`POST /api/storage-control-units/{storage-control-unit-id}/operations/undefine`
"Get Storage Control Unit Properties" on page 506	`GET /api/storage-control-units/{storage-control-unit-id}`
"Update Storage Control Unit Properties" on page 507	`POST /api/storage-control-units/{storage-control-unit-id}`
"Add Volume Range" on page 509	`POST /api/storage-control-units/{storage-control-unit-id}/operations/add-volume-range`
"Remove Volume Range" on page 511	`POST /api/storage-control-units/{storage-control-unit-id}/operations/remove-volume-range`

Table 96. DPM - Storage Control Unit: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Create Storage Path" on page 513	`POST /api/storage-control-units/{storage-control-unit-id}/storage-paths`
"Delete Storage Path" on page 516	`DELETE /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}`
"Get Storage Path Properties" on page 518	`GET /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}`
"Update Storage Path Properties" on page 519	`POST /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}`

Table 97. DPM - Storage Control Unit: URI variables	
Variable	**Description**
{storage-subsystem-id}	Object ID of a Storage Subsystem object
{storage-path-id}	Element ID of the Storage Path object
{storage-control-unit-id}	Object ID of a Storage Control Unit object

Storage Group operations summary

The following table provides an overview of the operations provided for Storage Group objects.

Table 98. DPM - Storage Group: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Groups" on page 544	`GET /api/storage-groups`
"Create Storage Group" on page 547	`POST /api/storage-groups`
"Delete Storage Group" on page 553	`POST /api/storage-groups/{storage-group-id}`
"Get Storage Group Properties" on page 556	`GET /api/storage-groups/{storage-group-id}`
"Modify Storage Group Properties" on page 558	`POST /api/storage-groups/{storage-group-id}/operations/modify`
"Resend Request" on page 568	`POST /api/storage-groups/{storage-group-id}/operations/resend-request`
"Add Candidate Adapter Ports to an FCP Storage Group" on page 571	`POST /api/storage-groups/{storage-group-id}/operations/add-candidate-adapter-ports`
"Remove Candidate Adapter Ports from an FCP Storage Group" on page 573	`POST /api/storage-groups/{storage-group-id}/operations/remove-candidate-adapter-ports`
"List Storage Volumes of a Storage Group" on page 575	`GET /api/storage-groups/{storage-group-id}/storage-volumes`
"Get Storage Volume Properties" on page 578	`GET /api/storage-groups/{storage-group-id}/storage-volumes{storage-volume-id}`
"Fulfill FICON Storage Volume" on page 580	`POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-ficon-storage-volume`

Table 98. DPM - Storage Group: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Fulfill FCP Storage Volume" on page 587	`POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/fulfill-fcp-storage-volume`
"Accept Mismatched Storage Volumes" on page 589	`POST /api/storage-groups/{storage-group-id}/operations/accept-mismatched-storage-volumes`
"Reject Mismatched FCP Storage Volumes" on page 591	`POST /api/storage-groups/{storage-group-id}/operations/reject-mismatched-storage-volumes`
"List Virtual Storage Resources of a Storage Group" on page 594	`GET /api/storage-groups/{storage-group-id}/virtual-storage-resources`
"Get Virtual Storage Resource Properties" on page 597	`GET /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}`
"Update Virtual Storage Resource Properties" on page 598	`POST /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}`
"Get Partitions for a Storage Group" on page 601	`GET /api/storage-groups/{storage-group-id}/operations/get-partitions`
"Validate LUN Path" on page 603	`POST /api/cpcs/{cpc-id}/operations/validate-lun-path`
"Start FCP Storage Discovery" on page 605	`POST /api/storage-groups/{storage-group-id}/operations/start-fcp-storage-discovery`
"Get Connection Report" on page 608	`GET /api/storage-groups/{storage-group-id}/operations/get-connection-report`
"Get Storage Group Histories" on page 617	`GET /api/console/operations/get-storage-group-histories`

Table 99. DPM - Storage Group: URI variables	
Variable	**Description**
{storage-group-id}	Object ID of a Storage Group object
{storage-volume-id}	Element ID of a Storage Volume element object
{virtual-storage-resource-id}	Element ID of a Virtual Storage Resource element object
{cpc-id}	Object ID of the CPC

Storage Template operations summary

The following table provides an overview of the operations provided for Storage Template objects.

Table 100. DPM - Storage Template: operations summary	
Operation name	**HTTP method and URI path**
"List Storage Templates" on page 642	`GET /api/storage-templates`
"Create Storage Template" on page 644	`POST /api/storage-templates`

Table 100. DPM - Storage Template: operations summary (continued)	
Operation name	**HTTP method and URI path**
“Delete Storage Template” on page 647	`DELETE /api/storage-templates/{storage-template-id}`
“Get Storage Template Properties” on page 649	`GET /api/storage-templates/{storage-template-id}`
“Modify Storage Template Properties” on page 650	`POST /api/storage-templates/{storage-template-id}/operations/modify`
“List Storage Template Volumes of a Storage Template” on page 656	`GET /api/storage-templates/{storage-template-id}/storage-template-volumes`
“Get Storage Template Volume Properties” on page 659	`GET /api/storage-templates/{storage-template-id}/storage-template-volumes/{storage-template-volume-id}`

Table 101. DPM - Storage Template: URI variables	
Variable	**Description**
{storage-template-id}	Object ID of a Storage Template object
{storage-template-volume-id}	Element ID of a Storage Template Volume element object

Tape Library operations summary

The following table provides an overview of the operations provided for Tape Library objects.

Table 102. DPM - Tape Library: operations summary	
Operation name	**HTTP method and URI path**
“List Tape Libraries” on page 663	`GET /api/tape-libraries`
“Undefine Tape Library” on page 665	`POST /api/tape-libraries/{tape-library-id}/operations/undefine`
“Get Tape Library Properties” on page 666	`GET /api/tape-libraries/{tape-library-id}`
“Update Tape Library Properties” on page 668	`POST /api/tape-libraries/{tape-library-id}`
“Request Tape Library Zoning” on page 670	`POST /api/tape-libraries/operations/request-tape-library-zoning`
“Discover Tape Libraries” on page 672	`POST /api/tape-libraries/operations/discover-tape-libraries`

Table 103. DPM - Tape Library: URI variables	
Variable	**Description**
{tape-library-id}	Object ID of a Tape Library object

Tape Link operations summary

The following table provides an overview of the operations provided for Tape Link objects.

Table 104. DPM - Tape Link: operations summary	
Operation name	**HTTP method and URI path**
"List Tape Links" on page 683	`GET /api/tape-links`
"Create Tape Link" on page 685	`POST /api/tape-links`
"Get Tape Link Properties" on page 689	`GET /api/tape-links/{tape-link-id}`
"Modify Tape Link Properties" on page 691	`POST /api/tape-links/{tape-link-id}/operations/modify`
"Delete Tape Link" on page 695	`POST /api/tape-links/{tape-link-id}/operations/delete`
"Add Adapter Ports" on page 697	`POST /api/tape-links/{tape-link-id}/operations/add-adapter-ports`
"Remove Adapter Ports" on page 700	`POST /api/tape-links/{tape-link-id}/operations/remove-adapter-ports`
"Replace Adapter Port" on page 702	`POST /api/tape-links/{tape-link-id}/operations/replace-adapter-port`
"Resend Request" on page 704	`POST /api/tape-links/{tape-link-id}/operations/resend-request`
"List Virtual Tape Resources of a Tape Link" on page 707	`GET /api/tape-links/{tape-link-id}/virtual-tape-resources`
"Get Virtual Tape Resource Properties" on page 709	`GET /api/tape-links/{tape-link-id}/virtual-tape-resources/{virtual-tape-resource-id}`
"Update Virtual Tape Resource Properties" on page 710	`POST /api/tape-links/{tape-link-id}/virtual-tape-resources/{virtual-tape-resource-id}`
"Get Partitions for a Tape Link" on page 713	`GET /api/tape-links/{tape-link-id}/operations/get-partitions`
"Get Tape Link Histories" on page 715	`POST /api/tape-links/{tape-link-id}/operations/get-tape-link-histories`
"Update Tape Link Environment Report" on page 726	`POST /api/tape-links/{tape-link-id}/operations/update-tape-link-environment-report`
"Get Tape Link Environment Report" on page 729	`GET /api/tape-links/{tape-link-id}/operations/get-tape-link-environment-report`

Table 105. DPM - Tape Link: URI variables	
Variable	**Description**
{tape-link-id}	Object ID of a Tape Link object
{virtual-tape-resource-id}	Object ID of a Virtual Tape Resource element object

Partition Link operations summary

The following table provides an overview of the operations provided for Partition Link objects.

Table 106. DPM - Partition Link: operations summary	
Operation name	**HTTP method and URI path**
"Create Partition Link" on page 742	`POST /api/partition-links`
"Delete Partition Link" on page 755	`POST /api/partition-links/{partition-link-id}/operations/delete`
"Get Partition Link Properties" on page 759	`GET /api/partition-links/{partition-link-id}`
"List Partition Links" on page 762	`GET /api/partition-links`
"Modify Partition Link" on page 764	`POST /api/partition-links/{partition-link-id}/operations/modify`

Table 107. DPM - Partition Link: URI variables	
Variable	**Description**
{partition-link-id}	Object ID of a Partition Link object

Partition object

The Partition object is central to partition management for IBM Dynamic Partition Manager (DPM). It stores configuration data and is the focal point for various DPM operations.

Data model

This object includes the properties defined in the "Base managed object properties schema" on page 100, including the operational-status-related properties, with the following class-specific specializations:

Table 108. Partition object: base managed object properties specializations			
Name	**Qualifier**	**Type**	**Description of specialization**
object-uri	—	String/URI	The canonical URI path for a Partition object is of the form `/api/partitions/{partition-id}`, where *{partition-id}* is the value of the **object-id** property of the Partition object.
parent	—	String/URI	The canonical URI path of the hosting CPC object.
class	—	String	The class of the Partition object is **"partition"**.
name	(w)(pc)	String (1-64)	The name of the partition. The name must be unique on a hosting CPC. The length and character requirements on this property are the same as those described in the "Base managed object properties schema" on page 100.
description	(w)(pc)	String (0-1024)	The description associated with this partition. Default: an empty string.

Table 108. Partition object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
status	(sc)	String Enum	The current operational status of the managed resource. Values: • **"communications-not-active"** - This status indicates that the HMC is not communicating with the SE. • **"status-check"** - This status indicates that the current status of the partition is unknown. This state normally indicates a communication issue between the SE and the CPC. • **"stopped"** - The partition is not running on the CPC. If the partition's **reserve-resources** property is **false**, the CPC memory, processor, and adapter resources defined for the partition are not currently allocated to it and are thus available for use by other partitions. • **"terminated"** - The partition was previously active on the CPC, but is no longer executing the operating system because all processors have been stopped by the operating system through some in-band action. This status may indicate that the operating system has been shut down or has "crashed". This status represents a condition between **"active"** and **"stopped"**. Although the partition is not executing the operating system, all CPC memory, processor, and adapter resources defined for it remain allocated to it. The `Stop Partition` action can be used to release the partition's non-reserved resources and place it in **"stopped"** status. • **"starting"** - A transitional status between **"stopped"** and **"active"** indicating that a `Start Partition` action has been initiated to put the partition into execution, but processing of that action has not yet completed. • **"active"** - The partition is up and running. • **"stopping"** - A transitional status between **"active"** and **"stopped"** indicating that a `Stop Partition` action has been initiated to stop execution of the partition and release its non-reserved resources, but the processing of that action has not yet completed. • **"degraded"** - The partition is active and one or more resources of the partition are in a degraded state and are not available for use. • **"reservation-error"** - The partition's **reserve-resources** property is **true** but one or more resources of the partition are in a degraded state and are not available for use. • **"paused"** - The partition was previously active on the CPC, but it is not currently executing the operating system because all processors have been stopped by the user through HMC or SE actions. Typically, partitions are temporarily placed in this status as part of performing diagnosis activities on the operating system in the partition. Although the partition is not currently executing the operating system, all CPC memory, processor, and adapter resources defined for it remain allocated to it. The HMC or SE UI actions can be used to resume execution of the operating system and place it in **"active"** status again. Alternatively, The `Stop Partition` action can be used to release the partition's non-reserved resources and place it in **"stopped"** status.
additional-status	—		This property is not provided.

Class specific additional properties

In addition to the properties defined in the base managed object, this object includes the following additional class-specific properties:

Table 109. Partition object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
type	—	String Enum	Defines the type of the partition. One of the following Values: • **"linux"** - the partition is intended for running a Linux operating system. • **"ssc"** - the partition will be running an IBM Secure Service Container appliance. • **"zvm"** - the partition is intended for running a z/VM operating system. The **"ssc"** type is only available with SE Version 2.13.1 or later that has MCL P00339.304 installed. Default: **"linux"**
short-name	(w)(pc)	String (1-8)	The short name must be 1-8 characters long, made up of uppercase alphanumeric characters, and have an alphabetic first character. The words PHYSICAL, REC, SYSTEM, and PRIM*nnnn* (where *nnnn* is a 4-digit number) are reserved and cannot be used. The short name is provided to the operating system running in the partition, for example by the STORE SYSTEM INFORMATION instruction. Default: Auto-generated Constraint: This property can only be updated when the partition's **status** is **"stopped"**. The **short-name** must be unique on a hosting CPC.

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
partition-id	(w)(pc)	String (2)	The partition ID must be a two-character hex number from 00 - 7F. When **autogenerate-partition-id** is **true**: • The user will not be able to set the value of **partition-id**, either on `Create Partition` or `Update Partition Properties`. • The value can be either **null** when the partition is not active and **reserve-resources** is **false**, or a value in the defined range. • A non-conflicting value is automatically generated and assigned when the partition is started or when **reserve-resources** is set to **true**. It is cleared when the partition is stopped and **reserve-resources** is set to **false**. When **autogenerate-partition-id** is **false**: • Uniqueness of the **partition-id** is not checked in the `Create Partition` or `Update Partition Properties` operations; however, during partition activation it must not contain the same value as any reserved or already active partition on the hosting CPC. • The value specified by the user must be in the defined range. • The value remains assigned to the partition unless changed by the user, or cleared as a consequence of changing the value of **autogenerate-partition-id** to **true**. Default: **null** Constraint: This property can only be updated when the partition's **status** is **"stopped"**.
autogenerate-partition-id	(w)(pc)	Boolean	Indicates if the **partition-id** is to be auto-generated. Default: **true** Constraint: This property can only be updated when the partition's **status** is **"stopped"**.
cpc-name	(a)	String	The **name** property of the partition's parent CPC object.
se-version	(a)	String	The **se-version** property of the partition's parent CPC object.
os-name	(pc)	String (0-8)	An operating system provided value, used to identify the operating system instance. The format of the value is operating system dependent. If not provided by the operating system or if the partition's **status** is **"stopped"**, an empty string is returned.

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
os-type	(pc)	String (0-8)	A human readable form of the operating system provided value for the type of the operating system active in this partition. If not provided by the operating system or if the partition **status** is **"stopped"**, an empty string is returned.
os-version	(pc)	String (0-32)	A human readable form of the operating system provided value for the version of the operating system active in this partition. If not provided by the operating system or if the partition **status** is **"stopped"**, an empty string is returned.
reserve-resources	(w)(pc)	Boolean	If true, resource reservation is enabled for this partition, and all physical resources backing the virtual resources configured for this partition are allocated and reserved, even when the partition is in **"stopped"** state. This guarantees that the partition start will not fail due to non-availability of resources. Default: **false**
degraded-adapters	(pc)	Array of String/URI	Array of URIs referring to I/O adapters (NIC, HBA, virtual function, crypto adapter, and virtual storage resource) attached to the partition that are degraded. Only used if the **status** property of the partition is **"degraded"**. If the partition has no degraded adapters, the array is empty.
processor-mode	(w)(pc)	String Enum	Defines how processors are allocated to the partition. One of the following values: • **"dedicated"** - All processors in the partition are to be exclusively available to this specific partition. • **"shared"** - All processors in the partition are to be shareable across partitions. Default: shared Constraint: This property can only be updated when the partition's **status** is **"stopped"**.
cp-processors	(w)(pc)	Integer	Defines the number of general purpose processors (CP) to be allocated for the partition. Default: 0 **Note:** Exactly one of the **cp-processors** and **ifl-processors** must have a value other than 0. Partitions can have only CPs or only IFLs but not a mix of both.

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ifl-processors	(w)(pc)	Integer	Defines the number of IFL processors to be allocated for the partition. Default: 0 **Note:** Exactly one of the **cp-processors** and **ifl-processors** must have a value other than 0. Partitions can have only CPs or only IFLs but not a mix of both.
ifl-absolute-processor-capping	(w)(pc)	Boolean	Indicates if absolute processor capping for **ifl-processors** is enforced. Absolute processor capping prevents this partition from using any more than the specified number of physical processors. Default: **false**
cp-absolute-processor-capping	(w)(pc)	Boolean	Indicates if absolute processor capping for **cp-processors** is enforced. Absolute processor capping prevents this partition from using any more than the specified number of physical processors. Default: **false**
ifl-absolute-processor-capping-value	(w)(pc)	Float	The amount of absolute capping applied to **ifl-processors**. Valid range: 0.01-255.00 in increments of 0.01. Default: 1.0
cp-absolute-processor-capping-value	(w)(pc)	Float	The amount of absolute capping applied to **cp-processors**. Valid range: 0.01-255.00 in increments of 0.01. Default: 1.0
ifl-processing-weight-capped	(w)(pc)	Boolean	Whether the processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. **true**: Indicates the IFL processor processing weight for the partition is capped. It represents the partition's maximum share of **ifl-processors** resources, regardless of the availability of excess IFL processor resources. **false**: Indicates the IFL processor processing weight for the partition is not capped. It represents the share of **ifl-processors** resources guaranteed to a partition when all IFL processor resources are in use. Otherwise, when excess IFL processor resources are available, the partition can use them if necessary. Default: **false**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
cp-processing-weight-capped	(w)(pc)	Boolean	Indicates whether the processing weight for general purpose processors is a limit or a target. **true**: Indicates that the general purpose processor processing weight for the partition is capped. It represents the partition's maximum share of **cp-processors**. **false**: Indicates that the general purpose processor processing weight for the partition is not capped. It represents the share of processor resources guaranteed to a partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the partition can use them if necessary. Default: **false**
minimum-ifl-processing-weight	(w)(pc)	Integer	Represents the minimum amount of IFL processor resources allocated to the partition. Valid range: 1-999 Default: 1
minimum-cp-processing-weight	(w)(pc)	Integer	Represents the minimum amount of general purpose processor resources allocated to the partition. Valid range: 1-999 Default: 1
initial-ifl-processing-weight	(w)(pc)	Integer	Defines the initial processing weight of IFL processors. Valid range: 1-999 Default: 100
initial-cp-processing-weight	(w)(pc)	Integer	Defines the initial processing weight of CP processors. Valid range: 1-999 Default: 100
current-ifl-processing-weight	(pc)	Integer	Defines the current IFL processing weight.
current-cp-processing-weight	(pc)	Integer	Defines the current CP processing weight.
maximum-ifl-processing-weight	(w)(pc)	Integer	Represents the maximum amount of IFL processor resources allocated to the partition. Valid range: 1-999 Default: 999

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
maximum-cp-processing-weight	(w)(pc)	Integer	Represents the maximum amount of shared general processor resources allocated to the partition. Valid range: 1-999 Default: 999
processor-management-enabled	(w)(pc)	Boolean	Indicates whether the processor management is enabled. Default: **false**
initial-memory	(w)(pc)	Integer	The initial amount of memory to assign to the partition when it is started, specified in MB. This value must be at least 4096 when the partition's **type** is **"ssc"**, and it must be less than or equal to the value of **maximum-memory**. Default: 1024 **Note:** If the value input by the user does not fall on an increment boundary, it is rounded off to the closest increment boundary.
reserved-memory	(pc)	Integer	The amount of reserved memory in MB, which equals **maximum-memory** minus **initial-memory**.
maximum-memory	(w)(pc)	Integer	The maximum size, specified in MB, to which the partition's memory allocation can be increased while the partition is running. This value must be greater than or equal to the value of **initial-memory**, and it must be no larger than the amount of entitled memory on the system. Default: 1024 Constraint: This property can only be updated when the partition's **status** is **"stopped"**. **Note:** If the value input by the user does not fall on an increment boundary, it is rounded off to the closest increment boundary.
auto-start	(pc)	Boolean	Indicates whether the partition is enabled for auto activation. If **true**, this partition is automatically started when the CPC is activated. Default: **false**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-device	(w)(pc)	String Enum	The type of device from which the partition is booted when it is started. Types: • **"storage-adapter"** - Boot from the HBA specified in **"boot-storage-device"**. • **"storage-volume"** - Boot from the storage volume specified in **"boot-storage-volume"**. • **"network-adapter"** - Boot from the NIC specified in **"boot-network-device"**. • **"ftp"** - Boot from the host specified in **"boot-ftp-host"**. • **"ftps"** - Boot from the host specified in **"boot-ftp-host"** using FTP Secure (FTPS). • **"sftp"** - Boot from the host specified in **"boot-ftp-host"** using Secure File Transfer Protocol (SFTP). • **"removable-media"** - Boot from HMC removable media specified in **"boot-removable-media"**. • **"iso-image"** - Boot from the ISO image previously mounted and associated with this partition. • **"none"** - Implies that the partition is not currently bootable. Default: **"none"** **Note:** Because a newly created partition does not yet have any HBAs, NICs, mounted ISO images, nor does it have a **storage-group** attached, , boot device options **"network-adapter"**, **"storage-adapter"**, **"iso-image"** and **"storage-volume"** are not valid values for the **boot-device** field as specified in the request body for the `Create Partition` operation. If the partition **type** is **"ssc"**, only **"none"** can be set as the **boot-device** when creating or updating a partition.
boot-network-device	(w)(pc)	String/ URI	Specifies the network device that shall be used for the network (PXE) boot. The value must point to a valid NIC URI on the same partition when **boot-device** mode is **"network-adapter"**. The **type** of the NIC must be either **"iqd"** or **"osd"**. The value can be set to either **null** or a valid URI when **boot-device** contains a value not relevant to this field. Default: **null**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-ftp-host	(w)(pc)	String, String/ IPv4 Address, String/ IPv6 Address	Host name or the IP address of the FTP server that shall be used for the FTP boot. The value must point to a valid host name or the IP address when **boot-device** mode is **"ftp"**, **"ftps"** or **"sftp"**. The value can be set to either **null** or a valid host name or IP address when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**. Default: **null**
boot-ftp-username	(w)(pc)	String	The user name for the account on the FTP server from which the boot image shall be retrieved. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**. Default: **null**
boot-ftp-password	(wo)(pc)	String	The password for the account on the FTP server from which the boot image shall be retrieved. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**.
boot-ftp-insfile	(w)(pc)	String	The path to the INS-file on the FTP server. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**. Default: **null**
boot-removable-media	(w)(pc)	String	Specifies the boot image or the CD/DVD or the USB media containing a bootable image. This must point to a fully-qualified path on the HMC. The value must point to a valid path name when **boot-device** mode is **"removable-media"**. The value can be set to either **null** or a valid path name when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"removable-media"**. Default: **null**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-removable-media-type	(w)(pc)	String Enum	Specifies the type of the removable media. Valid values are **"cdrom"** and **"usb"**. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"removable-media"**. Default: **null**
boot-timeout	(w)(pc)	Integer (60-600)	The time, in seconds, that is waited before an ongoing boot is aborted. This is applicable for all modes of **boot-device**. Default: 60
boot-storage-device	(w)(pc)	String/ URI	Specifies the HBA that shall be used for the partition to boot. The value has to point to a valid HBA URI on the partition when **boot-device** mode is **"storage-adapter"**. The value can be set to either **null** or a valid HBA URI when **boot-device** contains a value not relevant to this field. Default: **null**
boot-storage-volume	(w)(pc)	String/ URI	Specifies the volume that shall be used for the partition to boot. The value has to point to a valid URI of a storage volume of **type "boot"** contained in a storage group attached to the partition when **boot-device** is **"storage-volume"**. The value can be set to either **null** or a valid boot volume URI when **boot-device** contains a value not relevant to this field. Default: **null**
boot-logical-unit-number	(w)(pc)	String (1-16)	The hexadecimal logical unit number (LUN) representing the boot device. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"storage-adapter"**. Default: an empty string

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-world-wide-port-name	(w)(pc)	String (16)	The worldwide port name (WWPN) of the storage controller containing the target SCSI device to be used for boot, in hexadecimal. The value can be set to either null or a valid value when **boot-device** contains a value not relevant to this field. Required if **boot-device** is **"storage-adapter"**. Default: an empty string
boot-configuration[3]	(w)(pc)	String Enum	Specifies how to determine the boot configuration used for booting the operating system. Values: • **"selector"** - Uses the boot configuration specified in **boot-configuration-selector**. • **"automatic"** - The boot loader automatically searches for a boot configuration and uses the first valid boot configuration defined in the operating system. Available only when the associated CPC has feature **secure-boot-with-certificates**. The value must not be **null** when **boot-device** is **"storage-volume"**. The value can be set to either **null** or a valid value when **boot-device** contains a value not relevant to this field. If the value is not set when updating **boot-device** to **"storage-volume"**, the value is initialized automatically to **"selector"**. **Note:** Not applicable if **boot-storage-volume** points to a FICON storage volume and **boot-loader-mode** is **"channel-command-word"**. [Added by feature **secure-boot-with-certificates**]

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-configuration-selector	(w)(pc)	Integer	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system to be loaded. Whether and how this parameter is used to determine boot parameters depends on the operating system and its boot process. For Linux on IBM Z, for example, this parameter selects which of the operating system's pre-configured boot configurations is to be used, with the selected boot configuration in turn specifying parameters such as the kernel to be loaded, the kernel parameters to be used, or which disk is used as part of the boot process. Valid range: 0-30 Default: 0, which indicates that the operating system's default boot configuration should be used. **Note:** Not applicable if **"boot-storage-volume"** points to a FICON storage volume and **boot-loader-mode** is set to **"channel-command-word"**. [Updated by feature **secure-boot-with-certificates**]
boot-record-lba	(w)(pc)	String (1-16)	Specifies the logical block number, in hexadecimal, of the anchor point for locating the operating system on the SCSI disk from which the operating system is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. For Linux on IBM Z, for example, this parameter specifies the block number of the master boot record, which is usually the first block (block number 0) on the boot device. Default: 0, identifying the first block on the device. **Note:** Not applicable if **"boot-storage-volume"** points to a FICON storage volume.
boot-load-parameters	(w)(pc)	String (0-8)	Specifies parameters that are passed unmodified to the operating system boot process. The way in which these parameters are used depends on the operating system, but in general, these parameters are intended to be used to select an entry in the boot menu or the boot loader. Valid characters are 0-9, A-Z, @, $, #, blank ( ), and period (.). Default: an empty string

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-os-specific-parameters	(w)(pc)	String (0-256)	Specifies parameters that are passed unmodified to the loaded operating system as part of the boot process. The way in which these parameters are used depends on the operating system, but in general, these parameters are intended to specify boot-time configuration settings. For Linux on IBM Z, for example, this property can be used to specify kernel parameters. Default: an empty string **Note:** Not applicable if **"boot-storage-volume"** points to a FICON storage volume and **boot-loader-mode** is **"channel-command-word"**. [Updated by feature **secure-boot-with-certificates**]
boot-iso-image-name	(pc)	String	Name of the ISO image. This property is changed by the `Mount ISO Image` and `Unmount ISO Image` operations. This name must not be an empty string, and it must not contain any of the following characters: \,",<,>,|,:,&,$,*,/ Default: **null**
boot-iso-ins-file	(w)(pc)	String	INS file location within the ISO image. This property is changed by the `Mount ISO Image` and `Unmount ISO Image` operations. Default: **null**
secure-execution[2]	(pc)	Boolean	If true, Secure Execution for Linux is enabled. If false, Secure Execution for Linux is not enabled. If the partition's **status** is **"stopped"**, a null object is returned.
secure-boot[2]	(w)(pc)	Boolean	If true, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the partition. Partition start will fail if the signatures do not match. [Updated by feature **secure-boot-with-certificates**] This property is only valid for partitions with **type "linux"**. Beyond that, it is only applicable for partitions with **boot-device "storage-volume"**. On an Update request, this property is only allowed in the request body if the preconditions listed above are met. Default: **false**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-loader-mode[3]	(w)(pc)	String Enum	Specifies the boot loader mode used when booting from a FICON storage volume. Values: • **"channel-command-word"** - The default mode available on all SE-versions. This value may only be set if **secure-boot** is **false**. • **"list-directed"** - Available when the associated CPC has feature **secure-boot-with-certificates**. This mode supports secure boot for FICON storage volumes and allows additional boot settings. The storage volume must have the correct format to work in **"list-directed"** mode. The value must not be **null** when **boot-device** is **"storage-volume"** and **boot-storage-volume** points to a FICON storage volume. The value can be set to either **null** or a valid value when **boot-device** contains a value not relevant to this field or **boot-storage-volume** points to an FCP or NVMe storage volume. If the value is not set when updating **boot-device** and **boot-storage-volume** points to a FICON storage volume, the value is initialized automatically to **"channel-command-word"** if **secure-boot** is **false** or **"list-directed"** if **secure-boot** is **true**. [Added by feature **secure-boot-with-certificates**]
boot-record-location[3]	(w)(pc)	boot-record-location object	Specifies the location of the boot record on the FICON storage volume. If the value is set to **null**, the boot record location is derived from the volume label. The value can be set to either **null** or a valid value when **boot-device** contains a value not relevant to this field or **boot-storage-volume** points to a FCP or NVMe storage volume. Default: null **Note:** Not applicable if **boot-storage-volume** points to a FICON storage volume and **boot-loader-mode** is **"channel-command-word"**. [Added by feature **secure-boot-with-certificates**]
assigned-certificate-uris[3]	(pc)	Array of String/URI	Array of URIs referring to the certificates that are assigned to this partition. Default: an empty array [Added by feature **secure-boot-with-certificates**]
access-global-performance-data	(w)(pc)	Boolean	Indicates if global performance data authorization control is requested. Default: **false**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
permit-cross-partition-commands	(w)(pc)	Boolean	Indicates if cross partition commands authorization control is requested. Default: **false**
access-basic-counter-set	(w)(pc)	Boolean	Indicates if basic counter set authorization control is requested. Default: **false**
access-problem-state-counter-set	(w)(pc)	Boolean	Indicates if problem state counter set authorization control is requested. Default: **false**
access-crypto-activity-counter-set	(w)(pc)	Boolean	Indicates is crypto activity counter set authorization control is requested. Default: **false**
access-extended-counter-set	(w)(pc)	Boolean	Indicates if extended counter set authorization control is requested. Default: **false**
access-coprocessor-group-set	(w)(pc) or — if **se-version** is **"2.15.0"** or later	Boolean	Indicates if coprocessor group set authorization control is requested. **Note:** When the **se-version** property of the associated CPC is **"2.15.0"** or later, this property is not permitted on an `Update Partition Properties` operation, and its value is always **false**.
access-basic-sampling	(w)(pc)	Boolean	Indicates if basic CPU sampling authorization control is requested. Default: **false**
access-diagnostic-sampling	(w)(pc)	Boolean	Indicates if diagnostic sampling authorization control is requested. May only be true if **access-basic-sampling** is **true**. Default: **false**
permit-des-key-import-functions	(w)(pc)	Boolean	Enables/disables the importing of DES keys for the associated partition. Default: **true**
permit-aes-key-import-functions	(w)(pc)	Boolean	Enables/disables the importing of AES keys for the associated partition. Default: **true**
permit-ecc-key-import-functions[2]	(w)(pc)	Boolean	Enables/disables the importing of Elliptic Curve Cryptography (ECC) keys for the associated partition. Default: **true**

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
threads-per-processor	(pc)	Integer	The number of threads the operating system running in this partition can use, for each processor allocated. If the partition has never been activated, a value of 0 is returned. After the initial activation of the partition, this value is controlled by the SMT (Simultaneous Multi-Threading) setting in the OS.
virtual-function-uris	(pc)	Array of String/URI	Array of URIs referring to virtual functions associated with the partition. If the partition has no virtual functions, the array is empty. Default: an empty array
nic-uris	(pc)	Array of String/URI	Array of URIs referring to defined NICs (network adapters) attached to the partition. If the partition has no NICs, the array is empty. Default: an empty array
hba-uris	(pc)	Array of String/URI	Array of URIs referring to defined HBAs attached to the partition. If the partition has no HBAs, the array is empty. Default: an empty array
storage-group-uris	(pc)	Array of String/URI	Array of URIs referring to storage groups attached to the partition. If the partition has no attached storage groups, or if the DPM Storage feature is disabled, the array is empty. Default: an empty array
tape-link-uris	(pc)	Array of String/URI	Array of URIs referring to tape links attached to the partition. If the partition has no attached tape links, or if the DPM FCP Tape feature is disabled, the array is empty. Default: an empty array
crypto-configuration	(pc)	crypto-configuration object	Single instance of a crypto-configuration nested object. See "crypto-configuration object properties" on page 228. The `Increase Crypto Adapter Configuration` operation can be used to set the crypto configuration for the partition. Default: **null**
ssc-host-name[1]	(w)(pc)	String/Hostname	The Secure Service Container host name. This string meets the requirements of the String/Hostname data type with the following exceptions: • length is 1-64 characters • valid characters are a-z, A-Z, 0-9, period(.), hyphen(-), and underscore(_)

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ssc-boot-selection[1]	(w)(pc)	String Enum	Indicates whether to run the Secure Service Container appliance installer or the Secure Service Container appliance itself. One of: • **"installer"** - Boot the Secure Service Container appliance installer to install the Secure Service Container appliance and then start it. • **"appliance"** - Start the most recently installed Secure Service Container appliance and resume its execution from where it was when the partition was stopped. On an Update request, this property can be set from **"appliance"** to **"intaller"** only. Default: **"installer"**
ssc-ipv4-gateway[1]	(w)(pc)	String/ IPv4 Address	The default IPv4 Gateway to be used when there is at least one NIC configured in static IPv4 mode. Default: null
ssc-ipv6-gateway	(w)(pc)	String/ IPv6 Address	The default IPv6 Gateway to be used when there is at least one NIC configured in static IPv6 mode. Default: null
ssc-dns-servers[1]	(w)(pc)	Array of String/ IPv4 or IPv6 Address	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set. Default: An empty array
ssc-master-userid[1]	(w)(pc)	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and underscore(_). Default: null
ssc-master-pw[1]	(wo)(pc)	String (8-256)	The Secure Service Container master user password. Valid characters are: a-z, A-Z, 0-9, and !@#$ %^&*()_+{}|<>?-=. Default: null
available-features-list	—	Array of partition-feature-info objects	The list of optional features or behavior supported by this Partition. If the Partition has no optional features, then an empty array is provided.

Table 109. Partition object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
partition-link-uris[4]	(pc)	Array of String/ URI	Array of URIs referring to the partition links that this partition is part of. If the partition is not part of any partition links, the array is empty. This property is present in the Data Model when any of the following features is available: **dpm-smcd-partition-link-management** feature, **dpm-hipersockets-partition-link-management** feature, **dpm-ctc-partition-link-management** feature. Default: an empty array [Added by feature **dpm-smcd-partition-link-management**] [Added by feature **dpm-hipersockets-partition-link-management**] [Added by feature **dpm-ctc-partition-link-management**]

Notes:

[1]On a Get request, this property is returned only when **type** is **"ssc"**. On an Update request, this property can be updated only when **type** is **"ssc"**.

[2]This property is returned only when the SE version is 2.15.0 or later.

[3]This property is returned only when the associated CPC has feature **secure-boot-with-certificates**.

[4]This property is returned only when the associated SE version is 2.16.0 with the suitable MCL bundle, or a later SE version.

Table 110. partition-feature-info object properties		
Name	**Type**	**Description**
name	String Enum	The name of the feature. One of: • **"dpm-storage-management"** - Indicates that the Partition supports Storage Groups and FICON storage resources. FCP and FICON storage resources are defined in Storage Groups, which are then attached to this Partition. If the Partition does not have this feature, FCP storage resources are represented by HBAs, which must be directly attached to this Partition. **"dpm-fcp-tape-management"** - Indicates that the Partition supports Tape Libraries linked through FCP connections. FCP tape resources are defined in Tape Links, which are then attached to this CPC's partitions. • **"dpm-smcd-partition-link-management"** - Indicates that the Partition supports Partition Links via SMC-D connections. SMC-D Partition Links can be attached to partitions of this CPC. [Updated by feature **dpm-smcd-partition-link-management**] These features are inherited from the features with the same name on the hosting CPC object, and thus, are enabled only when the hosting CPCs have these features enabled and are disabled by default. See Chapter 6, “Features,” on page 103 for a list of operations that are affected for each of these features.
description	String	A brief description of the feature.
state	Boolean	Indicates if the feature is currently enabled (**true**) or disabled (**false**) for this Partition.

*Table 111. boot-record-location object properties [Added by feature **secure-boot-with-certificates**]*			
Name	**Qualifier**	**Type**	**Description**
cylinder	(w)	String (1-7)	The hexadecimal cylinder value where the boot record is located. The allowed value range is from 0-FFFFFFF
head	(w)	String (1)	The hexadecimal head value where the boot record is located. The allowed value range is from 0-F
record	(w)	String (1-2)	The hexadecimal record value where the boot record is located. The allowed value range is from 1-FF

crypto-configuration object properties

The crypto configuration of a partition represents the elements that are required to enable the partition to make use of crypto adapters. The configuration is a nested structure, containing two pieces of information:

- A set of crypto adapters that will be used by this partition, and

- A set of Crypto Domain Configuration objects. (See Table 113 on page 229.)

A crypto configuration that contains no crypto adapters and no crypto domain configurations is valid and is known as an *empty crypto configuration*. A non-empty configuration must contain at least 1 crypto adapter and at least 1 crypto domain configuration with an **access-mode** of **"control-usage"**.

Table 112. crypto-configuration nested object properties

Name	Type	Description
crypto-adapter-uris	Array of String/ URI	Array of URIs listing all crypto adapters that this partition can use.
crypto-domain-configurations	Array of crypto-domain-configuration objects	Array listing all crypto-domain-configuration objects for this partition. See Table 113 on page 229.

Table 113. crypto-domain-configuration nested object properties

Name	Type	Description
domain-index	Integer	Index value that identifies the domain to which this configuration applies. Minimum index is 0, maximum index depends on the CPC model.
access-mode	String Enum	Specifies the way in which the partition can use this domain. Valid values are: • **"control"** - The partition can load cryptographic keys into the domain, but it may not use the domain to perform cryptographic operations. • **"control-usage"** - The partition can load cryptographic keys into the domain, and it can use the domain to perform cryptographic operations.

Crypto configuration conflicts

A crypto configuration conflict occurs when the crypto configuration of two (or more) partitions:

1. Have one (or more) adapter(s) in common, and
2. Specified **"control-usage"** for one (or more) identical domain index(es).

No more than one of the partitions involved in a given crypto configuration conflict may be active or have reserved resources at any one point in time.

According to this definition, the crypto configuration of two partitions can have multiple conflicts (regarding different adapters and/or different domains).

It is also possible for a partition to be involved in conflicts with multiple other partitions. For example, Partition A has 3 crypto adapters in its configuration. Partition B has 2 of those and Partition C has the other one. Assuming they all have a control-usage domain in common, Partition A is now involved in a conflict with Partition B and a separate conflict with Partition C.

Such conflicts are only allowed for partitions that are in **"stopped"** state, and without reserved resources. That means the system will prevent the creation of conflicting crypto configuration for the set of active partitions, and the set of **"stopped"** partitions that have **reserve-resources** enabled.

Data model - Virtual Function element object

The following table contains the Virtual Function element object properties.

Table 114. Partition object - Virtual Function element properties			
Name	**Qualifier**	**Type**	**Description of specialization**
element-id	—	String (36)	The unique identifier for the virtual function instance.
element-uri	—	String/ URI	The canonical URI path for the virtual function is of the form `/api/partitions/{partition-id}/virtual-functions/{virtual-function-id}`, where *{partition-id}* is the **object-id** of the partition, and the *{virtual-function-id}* is the **element-id** of the virtual function.
parent	—	String/ URI	The URI path of the partition that hosts this virtual function.
class	—	String (16)	Always **"virtual-function"**.
name	(w)(pc)	String (1-64)	Name of the virtual function. The name must be unique among all virtual functions of the partition. The length and character requirements on this property are the same as those described in the "Base managed object properties schema" on page 100.
description	(w)(pc)	String (0-1024)	Description of the virtual function. Default: an empty string.
device-number	(w)(pc)	String (4)	Device number of the virtual function. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. Default: auto-generated. Constraint: This number must be unique across the device numbers of all other Virtual Function elements and all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition.
adapter-uri	(w)(pc)	String/ URI	The canonical URI path for the associated Accelerator adapter.
fid	—	Integer	Functional ID of the associated accelerator adapter identified in **adapter-uri**, or null if the partition is not active.

Data model - NIC element object

The following table contains the NIC element object properties.

Table 115. Partition object - NIC element object properties			
Name	**Qualifier**	**Type**	**Description of specialization**
element-id	—	String (36)	The unique identifier for the NIC within the scope of the partition.
element-uri	—	String/ URI	The canonical URI path for the NIC is of the form `/api/partitions/{partition-id}/nics/{nic-id}`, where *{partition-id}* is the **object-id** of the partition, and the *{nic-id}* is the **element-id** of the NIC.

Table 115. Partition object - NIC element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
parent	—	String/ URI	The URI path of the partition that hosts this NIC.
class	—	String (3)	Always **"nic"**.
name	(w)(pc)	String (1-64)	Name of the NIC. The name must be unique among all NICs of the partition. The length and character requirements on this property are the same as those described in the "Base managed object properties schema" on page 100.
description	(w)(pc)	String (0-1024)	Description of the NIC. Default: an empty string.
device-number	(w)(pc)	String (4)	Device number of the NIC. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. If **type** is **"osd"** a range of 3 device numbers will be allocated. If **type** is **"cna"**, the range of device numbers allocated is indicated by the **function-range** property. Default: auto-generated. Constraint: If **type** is **"roce"** or **"cna"**, this number must be unique across the device numbers of all other NIC elements of **type "roce"** or **"cna"** and all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition. If **type** is **"iqd"** or **"osd"**, this number must be unique across the device numbers of all other NIC elements of **type "iqd"** or **"osd"** and all instances of objects listed in "Channel-based device numbers" on page 197 of the partition.
network-adapter-port-uri	(w)(pc)	String/ URI	The canonical URI path for the associated Network Port element object. Only present when **type** is **"roce"** or **"cna"**.
virtual-switch-uri	(w)(pc)	String/ URI	The canonical URI path for the associated Virtual Switch object. Only present when **type** is **"osd"** or **"iqd"**. Constraint: If **type** is **"iqd"** and the Partition belongs to a CPC with API feature **dpm-hipersockets-partition-link-management** available, this property is not writable. [Updated by feature **dpm-hipersockets-partition-link-management**]
type	—	String Enum	The **type** of the NIC. The value of this property is derived implicitly from the backing adapter associated with this NIC on the `Create NIC` operation. Valid values are: • **"roce"** - RDMA over Converged Ethernet. • **"iqd"** - Internal Queued Direct. • **"osd"** - OSA Direct Express • **"cna"** - Cloud Network Adapter

Table 115. Partition object - NIC element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
ssc-management-nic	(w)(pc)	Boolean	Indicates that this NIC should be used as a management NIC for Secure Service Container to access the web interface. Can only be set to **true** if the partition's **type** is **"ssc"**. If the partition's **type** is **"ssc"**, there must be at least one management NIC defined before the partition can be started. If true, other parameters are required (at least **ssc-ip-address-type**) for this NIC. If true, only OSA or HiperSockets adapters can be selected as backing adapters. If the associated SE is version 2.13.1, the only valid port for an OSA backing adapter is port 0.
ssc-ip-address-type[1]	(w)(pc)	String Enum	Secure Service Container IP address type. Valid types are: • **"ipv4"** - Network is configured in static IPv4 mode • **"ipv6"** - Network is configured in static IPv6 mode • **"linklocal"** - Network is configured in Link Local mode • **"dhcp"** - Network is configured in DHCP mode.
ssc-ip-address[1]	(w)(pc)	String/ IPv4 Address or String/ IPv6 Address	The IP address of the Secure Service Container management web interface.
ssc-mask-prefix[1]	(w)(pc)	String	Network mask of the Secure Service Container management NIC. Either the mask is provided in bit notation, e.g. "/24" (both for IPv4 and IPv6), or in mask notation, e.g. "255.255.255.0" (IPv4 only).
vlan-id	(w)(pc)	Integer (1-4094)	The VLAN ID associated with this NIC. When the partition's **type** is **"ssc"**, this property is allowed only if the value of **ssc-management-nic** is **"true"**. It can be null. When the partition's **type** is not **"ssc"**, this property is not allowed when the **type** of the NIC is **"roce"** or **"cna"**.
mac-address	(w)(pc)	String (17)	The MAC address associated with this NIC. It must be unique among all the NICs created in the CPC. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addresses are valid, e.g. "02:ff:12:34:56:78". This value can not be set when the **type** of the NIC is **"roce"** or **"cna"**. Default: Auto-generated

Table 115. Partition object - NIC element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
vlan-type	(w)(pc)	String Enum	The type of VLAN tagging to use for the VLAN associated with this NIC, or null, if the NIC is not associated with a VLAN. Valid value: • **"enforced"** - the network adapter only allows untagged packets or packets tagged for the VLAN identified by **vlan-id** through to the operating system running in the partition. The network device in the operating system should also be configured with the same **vlan-id**. This value can not be set when the partition's **type** is **"ssc"** or when the **type** of the NIC is **"roce"** or **"cna"**.
function-number	(w)(pc)	Integer	The function number of the PCI function on the adapter. This property can only be set when the **type** of the NIC is **"cna"**. A value of 0 indicates a physical PCI function (PF); all other values indicate a virtual PCI function (VF). Default: 0
function-range	(w)(pc)	Integer (1-128)	The number of PCI functions on the NIC. This property can only be set when the **type** of the NIC is **"cna"**. It controls the number of virtual PCI functions that are created. For example, if **function-number** is 0 and **function-range** is 128, that results in 1 physical PCI function and 127 virtual PCI functions. Default: 128
[1]Only applicable if **ssc-management-nic** is **true**.			

Data model - HBA element object

An HBA represents a single Host Bus Adapter (HBA) available to a partition in a CPC. An HBA is an access point between a CPC and a Storage Area Network (SAN).

The following table contains the HBA element object properties.

Table 116. Partition object - HBA element object properties			
Name	**Qualifier**	**Type**	**Description of specialization**
element-uri	—	String/ URI	The canonical URI path of an HBA element is of the form `/api/partitions/{partition-id}/hbas/{hba-id}` where *{partition-id}* represents the **object-id** of the Partition object and *{hba-id}* represents the **element-id** of the HBA.
element-id	—	String (36)	The unique identifier for an HBA element. The string form of a UUID.
parent	—	String/ URI	The canonical URI path of the Partition object.
class	—	String	The class of an HBA element is **"hba"**.

Table 116. Partition object - HBA element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name of an HBA element. This name must be unique among all of the partition's HBA elements, and it must conform to the length and character requirements of the **name** property described in the "Base managed object properties schema" on page 100.
description	(w)(pc)	String (0-1024)	The description of the HBA element. Default: an empty string.
wwpn	—	String (16)	The worldwide port name of the HBA element. The string form of **wwpn** is of 16 hexadecimal characters.
device-number	(w)(pc)	String (4)	Device number of the HBA. The string is in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF. Default: auto-generated. Constraint: This number must be unique across the device numbers of all other HBA elements and all instances of the objects listed in "Channel-based device numbers" on page 197 of the partition.
adapter-port-uri	(pc)	String/ URI	The canonical URI path of the Storage Port element object to which this HBA is connected.

List Partitions of a CPC

The `List Partitions of a CPC` operation lists the partitions of a CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/partitions
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	A regular expression used to limit returned objects to those that have a matching **name** property.
`status`	String Enum	Optional	Filter string to limit returned objects to those that have a matching status property. Value must be a valid partition **status** property value.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid partition **type** property value.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties (**name**, **object-uri**, **status**, **type**). This is a list of comma-separated strings where each string is a property name defined in the Partition object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`partitions`	Array of partition-info objects	Array of nested partition-info objects, described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path of the Partition object.
`name`	String	The **name** property of the Partition object.
`status`	String Enum	The **status** property of the Partition object.
`type`	String Enum	The **type** property of the Partition object.

Description

This operation lists the partition objects that belong to a CPC. The object URI, display name, status, and type are provided for each.

If the **name** query parameter is specified, the returned list is limited to those partition objects that have a **name** property matching the specified filter pattern. If the **status** query parameter is specified, the returned list is limited to those partition objects that have a **status** property matching the specified filter value. If the **type** query parameter is specified, the returned list is limited to those partition objects that have a **type** property matching the specified filter value. If no query parameters are provided, no filtering is done.

An object is only included in the list if the API user has object-access permission for that object.

If the **additional-properties** query parameter is specified, the response body is enhanced with the additionally requested properties. The presence and value of each requested property is the same as it would be in the response body of a `Get Partition Properties` operation. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **additional-properties** query parameter is omitted, only the default properties are included in the response.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 235.

If the CPC is not in DPM mode, or there are no partitions defined to the CPC, or no partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Object-access permission to any Partition object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 235.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/93634ff4-0599-3f7d-b937-7673de7dfd0c/partitions?name=t.* HTTP/1.1
x-api-session: 2izurpik57ciomzst8z0q1vsqg2kuvfe9qxdja6irmbovo8z1c
```

Figure 44. List Partitions of a CPC: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Feb 2017 09:08:33 GMT
content-type: application/json;charset=UTF-8
content-length: 504
{
 "partitions":[
  {
   "name":"testpweights",
   "object-uri":"/api/partitions/2fa1f646-e9d9-11e6-a392-42f2e9cfe851",
   "status":"stopped",
   "type":"ssc"
  },
  {
   "name":"testVnic",
   "object-uri":"/api/partitions/bc09b56e-e88b-11e6-8715-42f2e9cfe851",
   "status":"active",
   "type":"linux"
  },
  {
   "name":"testCrypto",
   "object-uri":"/api/partitions/798167ba-ec4a-11e6-a040-42f2e9cfe851",
   "status":"stopped",
   "type":"linux"
  },
  {
   "name":"test_ah",
   "object-uri":"/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851",
   "status":"stopped",
   "type":"ssc"
  }
 ]
}
```

Figure 45. List Partitions of a CPC: Response

List Permitted Partitions

The `List Permitted Partitions` operation lists partitions to which the API user has object-access permission.

HTTP method and URI

```
GET /api/console/operations/list-permitted-partitions
```

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid partition **type** property value.
`status`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **status** property. Value must be a valid partition **status** property value.
`has-unacceptable-status`	Boolean	Optional	Filter string to limit returned objects to those that have a matching **has-unacceptable-status** property. Valid values are **true** and **false**.
`cpc-name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching **name** property.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties (**name**, **object-uri**, **type**, **status**, **has-unaccptable-status**, **cpc-name**, **cpc-object-uri**, **se-version**). This is a list of comma-separated strings where each string is a property name defined in the Partition object's data model. [Added by features **dpm-hipersockets-partition-link-management** and **dpm-ctc-partition-link-management**]

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`partitions`	Array of partition-info objects	Array of nested partition-info objects as described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
`name`	String	The **name** property of the Partition object.
`object-uri`	String/ URI	The **object-uri** property of the Partition object.
`type`	String Enum	The **type** property of the Partition object.

Field name	Type	Description
`status`	String Enum	The **status** property of the Partition object.
`has-unacceptable-status`	Boolean	The **has-unacceptable-status** property of the Partition object.
`cpc-name`	String	The **name** property of the partition's parent CPC object.
`cpc-object-uri`	String/ URI	The **object-uri** property of the partition's parent CPC object.
`se-version`	String	The **se-version** property of the partition's parent CPC object.

Description

This operation lists the Partition objects to which the API user has object-access permission. Some basic properties are provided for each partition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid partition **type** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have a **type** property matching the specified value. If the **type** parameter is omitted, no such filtering is performed.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid partition **status** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have a **status** property matching the specified value. If the **status** parameter is omitted, no such filtering is performed.

If the **has-unacceptable-status** query parameter is specified, the returned list is limited to those partitions that have a **has-unacceptable-status** property matching the specified value. If the **has-unacceptable-status** parameter is omitted, no such filtering is performed.

If the **cpc-name** query parameter is specified, the returned list is limited to those partitions whose parent CPC's **name** property matches the specified filter pattern. If the **cpc-name** parameter is omitted, no such filtering is performed.

If the **additional-properties** query parameter is specified, the response body is enhanced with the additionally requested properties. The presence and value of each requested property is the same as it would be in the response body of a Get Partition Properties operation. That is, it may be omitted or contain a special value such as null, -1, or an empty string, if a prerequisite condition is not met. If the **additional-properties** query parameter is omitted, only the default properties are included in the response.

A partition is included in the list only if the API user has object-access permission to that object. If there is a partition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no partitions known to the HMC or if no partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 237.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/list-permitted-partitions HTTP/1.1
x-api-session: 1hxko1kyziy64pcd9a9ot59ceb9jnh7vg55ylro930kubzgva5
```

Figure 46. List Permitted Partitions: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 27 Aug 2018 18:37:52 GMT
content-type: application/json;charset=UTF-8
content-length: 294
{
   "partitions":[
      {
         "cpc-name":"SEDPM005",
         "cpc-object-uri":"/api/cpcs/e4e18781-8063-3f2c-8222-044eb58988d9",
         "se-version":"2.14.0",
         "has-unacceptable-status":true,
         "name":"part1",
         "object-uri":"/api/partitions/592125be-76dd-11e7-94f9-02c2000226b7",
         "status":"communications-not-active",
         "type":"linux"
      }
   ]
}
```

Figure 47. List Permitted Partitions: Response

Usage note

The response body of this operation is similar to that of the `Get Inventory` operation, but it returns only a subset of partition properties. The response also includes some properties of the parent CPC, regardless of whether the API user has object-access permission to that CPC.

Create Partition

The `Create Partition` operation creates a partition with the given properties on the identified CPC.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/partitions
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`type`	String Enum	Optional	The value to be set as the partition's **type** property.
`name`	String (1-64)	Required	The value to be set as the partition's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the partition's **description** property.
`short-name`	String (8)	Optional	The value to be set as the partition's **short-name** property.
`partition-id`	String (2)	Required if **autogenerate-partition-id** is **false**	The value to be set as the partition's **partition-id** property.
`autogenerate-partition-id`	Boolean	Optional	The value to be set as the partition's **autogenerate-partition-id** property.
`ifl-processors`	Integer	Required if **cp-processors** is not provided	The value to be set as the partition's **ifl-processors** property.
`cp-processors`	Integer	Required if **ifl-processors** is not provided	The value to be set as the partition's **cp-processors** property.
`processor-mode`	String Enum	Optional	The value to be set as the partition's **processor-mode** property.
`initial-memory`	Integer	Required	The value to be set as the partition's **initial-memory** property.
`maximum-memory`	Integer	Required	The value to be set as the partition's **maximum-memory** property.
`reserve-resources`	Boolean	Optional	The value to be set as the partition's **reserve-resources** property.
`boot-device`	String Enum	Optional	The value to be set as the partition's **boot-device** property.
`boot-timeout`	Integer (60-600)	Optional	The value to be set as the partition's **boot-timeout** property.
`boot-ftp-host`	String	Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**	The value to be set as the partition's **boot-ftp-host** property.
`boot-ftp-username`	String	Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**	The value to be set as the partition's **boot-ftp-username** property.
`boot-ftp-password`	String	Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**	The value to be set as the partition's **boot-ftp-password** property.

Field name	Type	Rqd/Opt	Description
`boot-ftp-insfile`	String	Required if **boot-device** is **"ftp"**, **"ftps"** or **"sftp"**	The value to be set as the partition's **boot-ftp-insfile** property.
`boot-removable-media`	String	Required if **boot-device** is **"removable-media"**	The value to be set as the partition's **boot-removable-media** property.
`boot-removable-media-type`	String Enum	Required if **boot-device** is **"removable-media"**	The value to be set as the partition's **boot-removable-media-type** property.
`access-global-performance-data`	Boolean	Optional	The value to be set as the partition's **access-global-performance-data** property.
`permit-cross-partition-commands`	Boolean	Optional	The value to be set as the partition's **permit-cross-partition-commands** property.
`access-basic-counter-set`	Boolean	Optional	The value to be set as the partition's **access-basic-counter-set** property.
`access-problem-state-counter-set`	Boolean	Optional	The value to be set as the partition's **access-problem-state-counter-set** property.
`access-crypto-activity-counter-set`	Boolean	Optional	The value to be set as the partition's **access-crypto-activity-counter-set** property.
`access-extended-counter-set`	Boolean	Optional	The value to be set as the partition's **access-extended-counter-set** property.
`access-coprocessor-group-set`	Boolean	Optional	The value to be set as the partition's **access-coprocessor-group-set** property.
`access-basic-sampling`	Boolean	Optional	The value to be set as the partition's **access-basic-sampling** property.
`access-diagnostic-sampling`	Boolean	Optional	The value to be set as the partition's **access-diagnostic-sampling** property.
`permit-des-key-import-functions`	Boolean	Optional	The value to be set as the partition's **permit-des-key-import-functions** property.
`permit-aes-key-import-functions`	Boolean	Optional	The value to be set as the partition's **permit-aes-key-import-functions** property.
`permit-ecc-key-import-functions`	Boolean	Optional	The value to be set as the partition's **permit-ecc-key-import-functions** property.
`ssc-host-name`	String/ Hostname	Required, if **type** is **"ssc"**	The value to be set as the partition's **ssc-host-name** property.
`ssc-ipv4-gateway`	String/ IPv4 Address	Optional	The value to be set as the partition's **ssc-ipv4-gateway** property.

Field name	Type	Rqd/Opt	Description
`ssc-ipv6-gateway`	String/ IPv6 Address	Optional	The value to be set as the partition's **ssc-ipv6-gateway** property.
`ssc-dns-servers`	Array of String/ IPv4 or IPv6 Address	Optional	The value to be set as the partition's **ssc-dns-servers** property.
`ssc-master-userid`	String	Required, if **type** is **"ssc"**	The value to be set as the partition's **ssc-master-userid** property.
`ssc-master-pw`	String	Required, if **type** is **"ssc"**	The value to be set as the partition's **ssc-master-pw** property.
`initial-ifl-processing-weight`	Integer (1-999)	Optional	The value to be set as the partition's **initial-ifl-processing-weight** property.
`initial-cp-processing-weight`	Integer (1-999)	Optional	The value to be set as the partition's **initial-cp-processing-weight** property.
`acceptable-status`	Array of String Enum	Optional	The value to be set as the partition's **acceptable-status** property.
`cp-absolute-processor-capping`	Boolean	Optional	The value to be set as the partition's **cp-absolute-processor-capping** property.
`cp-absolute-processor-capping-value`	Float	Optional	The value to be set as the partition's **cp-absolute-processor-capping-value** property.
`cp-processing-weight-capped`	Boolean	Optional	The value to be set as the partition's **cp-processing-weight-capped** property.
`ifl-absolute-processor-capping`	Boolean	Optional	The value to be set as the partition's **ifl-absolute-processor-capping** property.
`ifl-absolute-processor-capping-value`	Float	Optional	The value to be set as the partition's **ifl-absolute-processor-capping-value** property.
`ifl-processing-weight-capped`	Boolean	Optional	The value to be set as the partition's **ifl-processing-weight-capped** property.
`maximum-cp-processing-weight`	Integer	Optional	The value to be set as the partition's **maximum-cp-processing-weight** property.
`maximum-ifl-processing-weight`	Integer	Optional	The value to be set as the partition's **maximum-ifl-processing-weight** property.
`minimum-cp-processing-weight`	Integer	Optional	The value to be set as the partition's **minimum-cp-processing-weight** property.
`minimum-ifl-processing-weight`	Integer	Optional	The value to be set as the partition's **minimum-ifl-processing-weight** property.
`processor-management-enabled`	Boolean	Optional	The value to be set as the partition's **processor-management-enabled** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the created partition.

Description

This operation creates a partition with the values specified on the identified CPC and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

Any properties identified as required must be included in the request body. Any properties identified as optional may be excluded from the request body; if an optional property is not found in the request body, its value will be set to its default value.

If the request body contents are valid, the partition is created on the target CPC and its properties are defined to their corresponding request body content's properties' values. If a property is omitted from the request body, its default value is used when creating the partition.

If the API user does not have action/task permission to the **New Partition** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{cpc-id}* does not identify a CPC object for which the API user has object-access permission.

If the request body contents fail to validate, a 400 (Bad Request) status code is returned. This may occur because the document fails to define a required property. This may also occur if the document fails to define a single valid partition, for instance defining a property with an invalid value (e.g. an **initial-memory** value less than zero, or a **name** that is already in use). If the status of the CPC is not valid (The valid states are **"active"**, "**service-required"**, **"degraded"**, **"exceptions"**), 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC identified by *{cpc-id}*.
- Action/task permission to the **New Partition** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 242.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 117. Create Partition: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A partition with the **name** or **short-name** specified in the request body already exists.
	15	The specified **access-diagnostic-sampling** value is not valid when **access-basic-sampling** is **false**.
	18	A property that is only valid for **type "ssc"** was provided, but the **type** is not **"ssc"**. The partition **type "ssc"** only allows **"none"** for **boot-device**.
	20	The **type "ssc"** is not supported by the targeted CPC. The property **permit-ecc-key-import-functions** is not supported by the targeted CPC.
	117	**boot-device** cannot be set to **"network-adapter"**, **"storage-adapter"**, **"storage-volume"**, or **"iso-image"** at the time of partition creation.
	118	There is an error in the fields related to the partition ID. One of: • **autogenerate-partition-id** is false and the **partition-id** specified in the request body is already in use. • **autogenerate-partition-id** is false and **partition-id** is not included in the request body. • **autogenerate-partition-id** is true and **partition-id** is included in the request body.
403 (Forbidden)	1	API user does not have action/task permission to the **New Partition** task.
404 (Not Found)	1	The CPC with object ID *{cpc-id}* does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	The operation cannot be performed because the CPC designated by the URI does not have a valid status. The valid states are **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	2	The operation cannot be performed because the CPC designated by the request URI is currently busy performing some other operation.
	5	The operation cannot be performed because the CPC designated by the request URI is currently not enabled for DPM.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The **reserve-resources** value is **true** but resources are not available to be reserved for this partition's use.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Table 117. Create Partition: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
Create Partition: Request
POST /api/cpcs/d49a116c-d938-3b87-ad7c-444752db1216/partitions HTTP/1.1
x-api-session: v6n1aljy1tmlsjq7ki955u0s4t7qr8xabmiu0pbpbadgq7fe
content-type: application/json
content-length: 117
{
   "name":"Partition",
   "cp-processors":3,
   "initial-memory":1024,
   "maximum-memory":2048,
   "processor-mode":"shared"
}
```

Figure 48. Create Partition: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/9cfdf912-89cf-11e5-8092-020000000056
cache-control: no-cache
date: Fri, 13 Nov 2015 06:26:42 GMT
content-type: application/json;charset=UTF-8
content-length: 69
{
   "object-uri":"/api/partitions/9cfdf912-89cf-11e5-8092-020000000056"
}
```

Figure 49. Create Partition: Response

Delete Partition

The `Delete Partition` operation deletes the identified partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}
```

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Description

This operation deletes the designated partition, which includes the following actions:

- The partition's HBAs are disassociated from their backing physical adapters and deleted.
- The partition's NICs are disassociated from their backing virtual switches and deleted.
- The partition's virtual functions are disassociated from their backing physical adapters and deleted.
- The partition is disassociated from the crypto adapters and crypto domains for which it was configured.

- The ISO image is deleted.
- An Inventory Change notification is emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the API user does not have action/task permission to the `Delete Partition` operation, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if status of either the partition or the CPC is not valid to perform the operation. A 409 (Conflict) status code is also returned if another operation targeting the partition is already underway.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Delete Partition** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 118. Delete Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden	1	API user does not have action permission to the **Delete Partition** task.
404 (Not Found)	1	The partition with the object ID *{partition-id}* does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"stopped"** or **"reservation-error"**).
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation. The valid states are **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	110	The operation cannot be performed as the partition is a member of a Capacity Group.
	112	The operation cannot be performed as the partition is targeted in a scheduled operation.
	113	The operation cannot be performed as the partition is configured to be automatically started when its hosting CPC starts.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
DELETE /api/partitions/9cfdf912-89cf-11e5-8092-020000000056 HTTP/1.1
x-api-session: 515ad9soju9cvqsyxkcq3d65c1v2bd6e8rhz4pu2psps2jae1f
```

Figure 50. Delete Partition: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 13 Nov 2015 06:30:12 GMT

<No response body>
```

Figure 51. Delete Partition: Response

Usage note

This is a synchronous operation and as such does not complete until the partition has been deleted. Depending on the I/O configuration associated with the partition, this operation may take a considerable amount of time to complete. API clients that are concerned about that should use the `Delete Partition Asynchronously` operation instead.

Delete Partition Asynchronously

The `Delete Partition Asynchronously` operation deletes the identified partition asynchronously.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/async-delete
```

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Response body contents

Once the delete request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body. The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in “Job status and reason codes” on page 249. The **job-results** field contains null when the operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation asynchronously deletes the designated partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body contains a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 249.

The deletion includes the following actions:

- The partition's HBAs are disassociated from their backing physical adapters and deleted.
- The partition's NICs are disassociated from their backing virtual switches and deleted.
- The partition's virtual functions are disassociated from their backing physical adapters and deleted.
- The partition is disassociated from the crypto adapters and crypto domains for which it was configured.
- The ISO image is deleted.
- An Inventory Change notification is emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object for which the API user has object-access permission. If the API user does not have authority to perform the `Delete Partition` operation, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the status of either the partition or the CPC is not valid to perform the operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Delete Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 247.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 119. Delete Partition Asynchronously: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden	1	API user does not have the required permission for this operation.
404 (Not Found)	1	The **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.

Table 119. Delete Partition Asynchronously: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"stopped"** or **"reservation-error"**).
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 120. Delete Partition Asynchronously: Job status and reason codes		
HTTP error status code	**Reason code**	**Description**
204 (No Content)	N/A	Delete completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"stopped"** or **"reservation-error"**).
	2	The operation canot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	110	The operation cannot be performed as the partition is a member of a Capacity Group.
	112	The operation cannot be performed as the partition is targeted in a scheduled operation.
	113	The operation cannot be performed as the partition is configured to be automatically started when its hosting CPC starts.
500 (Server Error)	100	Partition delete failed.
	101	Partition delete job timed out.

Example HTTP interaction

```
POST /api/partitions/279cee22-50d3-11e7-b285-fa163ef98b8b/operations/async-delete HTTP/1.1
x-api-session: 37o1ukpoqdqwa64iggv0gjtfx6q9xsdihiquuidnvdh5s8twc9
content-type: application/json
```

Figure 52. Delete Partition Asynchronously: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/3ae9a3c8-55b3-11e7-b883-fa163e6e13bb
cache-control: no-cache
date: Tue, 20 Jun 2017 12:23:13 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/3ae9a3c8-55b3-11e7-b883-fa163e6e13bb"
}
```

Figure 53. Delete Partition Asynchronously: Response

Get Partition Properties

The `Get Partition Properties` operation retrieves the properties of a single Partition object.

HTTP method and URI

```
GET /api/partitions/{partition-id}
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Query parameters:

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the Partition object's data model.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Partition object as defined in the "Data model" on page 209. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

The `Get Partition Properties` operation returns the current values of the properties for the Partition object as defined in the "Data model" on page 209.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the properties query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined in the "Data model" on page 209. A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object designated by {partition-id}

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 250.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partitions/579e3d52-4492-11ed-9da8-fa163e9d3300 HTTP/1.1
x-api-session: 1drzqsy7qhxqmvasok616bm3he5k0zlvv72j4u93hf75070ycf
```

Figure 54. Get Partition Properties: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Wed, 05 Oct 2022 12:26:31 GMT
Content-Type: application/json;charset=UTF-8
Content-Length: 3590
{
   "acceptable-status":[
      "active"
   ],
   "access-basic-counter-set":false,
   "access-basic-sampling":false,
   "access-coprocessor-group-set":false,
   "access-crypto-activity-counter-set":false,
   "access-diagnostic-sampling":false,
   "access-extended-counter-set":false,
   "access-global-performance-data":false,
   "access-problem-state-counter-set":false,
   "assigned-certificate-uris":[
      "/api/certificates/7f0269ea-4492-11ed-b798-fa163e8f984a"
   ],
   "auto-start":false,
   "autogenerate-partition-id":true,
   "available-features-list":[
      {
         "description":"The DPM storage management approach in which FCP and FICON storage
            resources are defined in Storage Groups, which are attached to Partitions.",
         "name":"dpm-storage-management",
         "state":true
      },
      {
         "description":"The DPM enhancement to support FCP tape.",
         "name":"dpm-fcp-tape-management",
         "state":true
      },
      {
         "description":"The DPM enhancement to support SMC-D based partition links",
         "name":"dpm-smcd-partition-link-management",
         "state":true
      }
   ],
   "boot-configuration":"automatic",
   "boot-configuration-selector":0,
   "boot-device":"storage-volume",
   "boot-ftp-host":null,
   "boot-ftp-insfile":null,
   "boot-ftp-username":null,
   "boot-iso-image-name":null,
   "boot-iso-ins-file":null,
   "boot-load-parameters":"",
   "boot-loader-mode":"list-directed",
   "boot-logical-unit-number":"",
   "boot-network-device":null,
   "boot-os-specific-parameters":"",
   "boot-record-lba":"0",
   "boot-record-location":null,
   "boot-removable-media":null,
   "boot-removable-media-type":null,
   "boot-storage-device":null,
   "boot-storage-volume":"/api/storage-groups/4148d0a0-4490-11ed-b90d-fa163e9d3300/
      storage-volumes/42b0427a-4490-11ed-b90d-fa163e9d3300",
   "boot-timeout":60,
   "boot-world-wide-port-name":"",
   "class":"partition",
   "cp-absolute-processor-capping":false,
   "cp-absolute-processor-capping-value":1.0,
   "cp-processing-weight-capped":false,
   "cp-processors":2,
   "cpc-name":"5X100635",
   "crypto-configuration":null,
```

Figure 55. Get Partition Properties: Response (Part 1)

```
    "current-cp-processing-weight":1,
    "current-ifl-processing-weight":1,
    "degraded-adapters":[],
    "description":"An example Partition coming with a NIC, a SMC-D partition link,
       and a FICON storage group; configured for secure boot from SAN.",
    "has-unacceptable-status":true,
    "hba-uris":[],
    "ifl-absolute-processor-capping":false,
    "ifl-absolute-processor-capping-value":1.0,
    "ifl-processing-weight-capped":false,
    "ifl-processors":0,
    "initial-cp-processing-weight":100,
    "initial-ifl-processing-weight":100,
    "initial-memory":6144,
    "is-locked":false,
    "maximum-cp-processing-weight":999,
    "maximum-ifl-processing-weight":999,
    "maximum-memory":10240,
    "minimum-cp-processing-weight":1,
    "minimum-ifl-processing-weight":1,
    "name":"ex1",
    "nic-uris":[
       "/api/partitions/579e3d52-4492-11ed-9da8-fa163e9d3300/nics/
          5a4faa04-4492-11ed-9da8-fa163e9d3300"
    ],
    "object-id":"579e3d52-4492-11ed-9da8-fa163e9d3300",
    "object-uri":"/api/partitions/579e3d52-4492-11ed-9da8-fa163e9d3300",
    "os-current-cp-processors":0,
    "os-current-ifl-processors":0,
    "os-current-memory":0,
    "os-name":"",
    "os-type":"",
    "os-version":"",
    "parent":"/api/cpcs/6fa255ad-db6d-3112-b9a5-54b3e7f81063",
    "partition-id":null,
    "partition-link-uris":[
       "/api/partition-links/6ab4567c-4494-11ed-a24e-fa163e9d3300"
    ],
    "permit-aes-key-import-functions":true,
    "permit-cross-partition-commands":false,
    "permit-des-key-import-functions":true,
    "permit-ecc-key-import-functions":true,
    "processor-management-enabled":false,
    "processor-mode":"shared",
    "reserve-resources":false,
    "reserved-memory":4096,
    "se-version":"2.16.0",
    "secure-boot":true,
    "secure-execution":false,
    "short-name":"EX1",
    "status":"stopped",
    "storage-group-uris":[
       "/api/storage-groups/4148d0a0-4490-11ed-b90d-fa163e9d3300"
    ],
    "tape-link-uris":[],
    "threads-per-processor":1,
    "type":"linux",
    "virtual-function-uris":[]
}
```


Figure 56. Get Partition Properties: Response (Part 2)

Update Partition Properties

The `Update Partition Properties` operation updates one or more of the writable properties of the Partition object designated by *{partition-id}*.

HTTP method and URI

```
POST /api/partitions/{partition-id}
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain one or more field names representing writable partition properties, along with the new values for those fields. The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Partition object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type. If the API user does not have action/task permission to the `Partition Details` operation, a 403 (Forbidden) status code is returned. If the status of the CPC hosting the partition is not in a valid state, 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition is in a transitional state (**"starting"** or **"stopping"**) or if user sets **boot-device** to **"iso-image"**, but there is no ISO image mounted on the partition.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 121. Update Partition Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected.
	8	A partition with the **name** or **short-name** specified in the request body already exists.
	15	**access-diagnostic-sampling** cannot be **true** when **access-basic-sampling** is **false**. **boot-device** setting is invalid. Since **secure-boot** is set to **true**, only **boot-device "storage-volume"** is allowed.
	18	A property that is only valid for **type "ssc"** was provided, but the **type** is not **"ssc"**. **type "ssc"** only allows **"none"** for **boot-device**.
	19	The request body contains a field whose corresponding data model property is not writable on this HMC and/or SE version.
	118	There is an error in the fields related to the partition ID. One of: • **autogenerate-partition-id** is false and the **partition-id** specified in the request body is already in use. • **autogenerate-partition-id** is false and **partition-id** is not included in the request body. • **autogenerate-partition-id** is true and **partition-id** is included in the request body.
403 (Forbidden)	1	API user does not have the required permission for this operation
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.
	2	A URI in the request body does not designate a resource of an expected type.

Table 121. Update Partition Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	8	The operation cannot be completed because it would result in inconsistencies between the **boot-device** property and the related properties (**boot-device**, **boot-loader-mode**, **secure-boot**) [Updated by feature **secure-boot-with-certificates**]. **access-diagnostic-sampling** cannot be set to **true** when **access-basic-sampling** is **false**. **access-basic-sampling** cannot be set to **false** when **access-diagnostic-sampling** is **true.** **secure-boot** cannot be set to **true** unless the **type** is **"linux"**, the **boot-device** is **"storage-volume",** and the volume type is **fcp.**
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	110	Partition's **processor-mode** cannot be updated as the partition is a member of a Capacity Group.
	116	The **reserve-resources** value is **true** but resources are not available to be reserved for this partition's use.
	119	The value of the **boot-storage-volume** property does not designate a storage volume of **type "boot"** contained in a storage group that is attached to the partition.
	120	• Request contains **boot-storage-volume** property when the **"dpm-storage-management"** feature is disabled. • **"storage-volume"** is provided as value for **boot-device** property when the **"dpm-storage-management"** feature is disabled. • Request contains **ipl-load-parameter** when the **"dpm-storage-management"** feature is disabled.
	121	• Request contains **boot-storage-device** property when the **"dpm-storage-management"** feature is enabled. • **"storage-adapter"** is provided as a value for **boot-device** property when the **"dpm-storage-management"** feature is enabled. • Request contains **boot-logical-unit-number** when the **"dpm-storage-management"** feature is enabled. • Request contains **boot-world-wide-port-name** when the **"dpm-storage-management"** feature is enabled.

Table 121. Update Partition Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	122	The **storage-volume** provided as value for **boot-storage-volume** property is **"incomplete"** or is not mapped to any LUN.
	125	One or more domains of **type "control-usage"** could not be removed from the crypto configuration because the designated partition is active and the corresponding crypto configuration includes one ore more crypto adapters in state **"online"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/30d8fe00-89c3-11e5-9b53-020000000056 HTTP/1.1
x-api-session: 2b1cv9k5i0v3tfgm6uo5vggzzj3eoer6zeiu7jt79tznjeqanl
content-type: application/json
content-length: 90
{
   "cp-processors":5,
   "description":"Sample partition description",
   "name":"Serv-sample"
}
```

Figure 57. Update Partition Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 13 Nov 2015 06:43:45 GMT

<No response body>
```

Figure 58. Update Partition Properties: Response

Usage note

This is a synchronous operation and as such does not complete until the partition has been updated. Depending on the I/O configuration associated with the partition, this operation may take a considerable amount of time to complete. API clients that are concerned about that should use the `Update Partition Properties Asynchronously` operation instead.

Update Partition Properties Asynchronously

The `Update Partition Properties Asynchronously` operation updates one or more of the writable properties of the Partition object designated by *{partition-id}* asynchronously.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/async-update
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain one or more field names representing writable partition properties, along with the new values for those fields. The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Response body contents

Once the update request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body. The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 259. The **job-results** field contains null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The request body object is validated against the data model for the Partition object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

This operation asynchronously updates the designated partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body contains a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 259.

On successful execution of the asynchronous portion of this operation, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned in the asynchronous result document. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type. If the API user does not have action/task permission to the `Partition Details` operation, a 403 (Forbidden) status code is returned. If the status of the CPC hosting the partition is not in a valid state, 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition is in a transitional state (**"starting"** or **"stopping"**).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 258.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 122. Update Partition Properties Asynchronously: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 123. Update Partition Properties Asynchronously: Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Update completed successfully.

Table 123. Update Partition Properties Asynchronously: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected.
	8	A partition with the **name** or **short-name** specified in the request body already exists.
	15	**access-diagnostic-sampling** cannot be **true** when **access-basic-sampling** is **false**.
	18	A property that is only valid for **type "ssc"** was provided, but the **type** is not **"ssc"**. The partition **type "ssc"** only allows **"none"** for the **boot-device**.
	118	There is an error in the fields related to the partition ID. One of: • **autogenerate-partition-id** is false and **partition-id** specified in the request body is already in use. • **autogenerate-partition-id** is false and **partition-id** is not included in the request body. • **autogenerate-partition-id** is true and **partition-id** is included in the request body.

Table 123. Update Partition Properties Asynchronously: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	8	**boot-device** can only be set if the boot device type (**"network-adapter"**, **"storage-adapter"**, **"ftp"**, **"sftp"**, **"ftps"**, **"iso-image"**, etc.) points to a valid URI/Hostname/ISO image. Boot device type cannot be set to null if the boot device points to the corresponding boot device type. [Updated by feature **secure-boot-with-certificates**] **access-diagnostic-sampling** cannot be set to true when **access-basic-sampling** is false. **access-basic-sampling** cannot be set to false when **access-diagnostic-sampling** is true.
	110	Partition's **processor-mode** cannot be updated as the partition is a member of a Capacity Group.
	116	The **reserve-resources** boolean is true but resources are not available to be reserved for this partition’s use.
	119	The value of the **boot-storage-volume** property does not designate a storage volume of **type "boot"** contained in a storage group that is attached to the partition.
	120	• Request contains **boot-storage-volume** property when the **"dpm-storage-management"** feature is disabled. • **"storage-volume"** is provided as value for **boot-device** property when the **"dpm-storage-management"** feature is disabled. • Request contains **ipl-load-parameter** when the **"dpm-storage-management"** feature is disabled.
	121	• Request contains **boot-storage-device** property when the **"dpm-storage-management"** feature is enabled. • **"storage-adapter"** is provided as a value for **boot-device** property when the **"dpm-storage-management"** feature is enabled. • Request contains **boot-logical-unit-number** when the **"dpm-storage-management"** feature is enabled. • Request contains **boot-world-wide-port-name** when the **"dpm-storage-management"** feature is enabled.
	122	The **storage-volume** provided as value for **boot-storage-volume** property is **"incomplete"** or is not mapped to any LUN.
500 (Server Error)	100	Partition update failed.
	101	Partition update job timed out.

Example HTTP interaction

```
POST /api/partitions/279cee22-50d3-11e7-b285-fa163ef98b8b/operations/async-update HTTP/1.1
x-api-session: 1wb0irgy7jn2edj7ptyqy4twit24m0b7krskctnf7h5dr91goz
content-type: application/json
content-length: 20
{
   "name":"Testname"
}
```

Figure 59. Update Partition Properties Asynchronously: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/df6b825a-55b2-11e7-8f25-fa163e6e13bb
cache-control: no-cache
date: Tue, 20 Jun 2017 12:20:39 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/df6b825a-55b2-11e7-8f25-fa163e6e13bb"
}
```

Figure 60. Update Partition Properties Asynchronously: Response

Start Partition

The `Start Partition` operation allocates the physical resources required by the partition and begins its execution on the CPC by booting the partition as configured by its boot-related properties.

HTTP method and URI

`POST /api/partitions/`*`{partition-id}`*`/operations/start`

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Depending on the **boot-device** property of the corresponding partition, the boot process might be handled by the zBootLoader component. The zBootLoader is used with partitions whose associated Support Element version is 2.15.0 or later:

- when **boot-device** is of **type "network-adapter"**, or
- when **boot-device** is of **type "storage-volume"** and **boot-storage-volume** refers to a volume of type FCP.

In case the zBootLoader fails to boot the corresponding partition, the error information provided by the zBootLoader is surfaced to users of the `Start Partition` operation through details of the corresponding **job-results** field (within the response body of the asynchronous job result):

- The **message** field contains an error message and message ID provided by the zBootLoader.

- The **bootloader-error-id** contains the error ID provided by the zBootLoader.

Notes:

- The zBootLoader might raise a console hardware message with additional information for further analysis.
- The zBootLoader is also used for partition **type "ssc"**, but in this case, the zBootLoader error information is not provided.

The set of zBootLoader error IDs are documented in the *Small Computer Systems Interface (SCSI) IPL - Machine Loader Messages*.

Once the start operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body from the `Start Partition` request.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 264. The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.
bootloader-error-id	String	If the zBootLoader failed to boot the partition, this field contains the zBootLoader error ID; otherwise, this field is not present.

Description

This operation asynchronously starts the identified partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 264.

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

If the Partition object's **boot-device** property is **"none"**, then no program is started in the new partition. However, other portions of the operation are performed and the partition is placed in the **"paused"** state.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the user does not have action/task permission to the `Start Partition` action, a 403 (Forbidden) status code is returned. If the status of the partition is not valid, 409 (Conflict) status code is returned. The valid states of the partition are **"stopped"** and **"reservation-error"**. A 409 (Conflict) status code is also returned if the CPC hosting the partition is not in a valid state, or if the CPC does not have sufficient processors, memory or adapter resources to allocate to the partition.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission for the **Start Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 262.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 124. Start Partition: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	Start operation completed successfully.

Table 124. Start Partition: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be one of the following states: **"active"**, **"service-required"**, **"degraded"** and **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	43	The operation has been canceled.
	110	There is no DHCP info or an error occurred when trying to resolve DHCP information.
	111	DHCP file error occurred.
	112	DNS lookup error has occurred.
	113	An error has occurred during network boot component/server/configuration download.
	114	An error occurred when parsing network boot configuration, or when executing network boot program.
	114	An error occurred during removable media load.
	115	An error occurred during FTP load.
	116	The CPC does not have sufficient processor, memory, or adapter resources to allocate to the partition to perform start operation.
	117	An error occurred during initialization on the network boot device or there was an error in internal setup during network boot.
	118	The count of the partitions in active state has reached its maximum.
	119	The specified device does not contain a bootable dump program.
	119	DHCP lease failed on device or there was a DHCP lease internal error.
	120	The partition configuration is not valid for a Secure Service Container partition. The specific error reason is returned as additional error text. Possible error conditions are: • Secure Service Container Management NIC is missing (there needs to be at least one Secure Service Container Management NIC). • At least one NIC was configured in static IPv4 mode, but no **ssc-ipv4-gateway** was provided.
	131	An error occurred in the PR/SM hypervisor during partition start, caused by invalid settings or configuration related problems (e.g., in the *.INS file).

Table 124. Start Partition: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	122	The operation cannot be performed because the **boot-storage-volume** property does not designate a storage volume that is fulfilled.
	125	A timeout occurred during partition boot. Check the integrity of the boot source, and the **boot-timeout** property.
	130	An error occurred during partition boot through the zBootLoader, caused by invalid settings or configuration related problems.
	131	An error occurred in the PR/SM hypervisor during partition start, caused by invalid settings or configuration related problems (e.g. in the *.INS file).
500 (Server Error)	100	Partition start failed.
	101	Partition start job timed out.
	130	An internal error occurred during partition boot through the zBootLoader.
	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/start HTTP/1.1
x-api-session: pd0nrulei0qsa1mwlpaw7cmq26rnsdcdhtp4w4m9gzse7gybg
content-type: application/json
```

Figure 61. Start Partition: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/913b0490-d537-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/913b0490-d537-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 62. Start Partition: Response

Attach Storage Group to Partition

The `Attach Storage Group to Partition` operation attaches a storage group to a partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/attach-storage-group
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
`storage-group-uri`	String/ URI	Required	The canonical URI of the Storage Group object to be attached to the partition.

Description

The `Attach Storage Group to Partition` operation attaches the storage group to the partition specified by the *{partition-id}* portion of the request URI.

On successful execution, the storage group gets associated with the partition.

For a storage group of type **"fc"**, the virtual storage resources are created if the partition is in one of the following states: **"active"**, **"degraded"**, **"paused"** or **"terminated"**, if the fulfillment state of the storage group is **"complete"**.

The virtual storage resources are created immediately on the storage ports depending on the status of the unassigned worldwide port names of the storage group. If the status of the worldwide port name is **"validated"**, a previously verified storage port is used to create the virtual storage resource. If the status of the worldwide port name is **"not-validated"**, a virtual storage resource is created without a storage port. In that case, a storage port is assigned to the virtual storage resource later, when DPM successfully verifies connectivity to the defined storage volumes on storage ports.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

This operation enables the operating system in the partition to access the storage volumes defined in the storage group, through the virtual storage resources that are assigned to storage ports. The operating system may not be able to access all the volumes defined in the storage group, until the fulfillment state of the storage group is **"complete"**.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **storage-group-uri** field in the request body must designate an existing Storage Group and the API user must have object-access permission to that storage group.

If either of these conditions are not met, status code 404 (Not Found) is returned. In addition, the API user must have action/task permissions to **Partition Details** task; otherwise, status code 403 (Forbidden) is returned. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the Storage Group object designated by the **storage-group-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 125. Attach Storage Group to Partition: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the storage group URI in request body field **storage-group-uri** does not designate an existing Storage Group object, or the API user does not have object-access permission to that storage group.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	13	The operation is not supported when the **"dpm-storage-management"** feature is not enabled on the partition.
	118	The Storage Group specified by **storage-group-uri** is already attached to the partition.
	119	The Storage Group specified by **storage-group-uri**is already attached to maximum partitions according to its specification.
	124	The storage group object designated by the *{storage-group-uri}* was busy performing some other operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/acab6d72-8107-11e8-9d6f-00106f0d81cb/operations/attach-storage-
   group HTTP/1.1
content-length:81,
content-type:application/json,
x-api-session:1zpqlegsp02sdzr2h04uxhc91ou8fdmxvpn0ukxtaouqg6k3e0
{
    "storage-group-uri": "/api/storage-groups/519578c6-9569-11e8-a732-00106f0d81cb"
}
```

Figure 63. Attach Storage Group to Partition: Request

```
204 No Content
cache-control:no-cache,
date:Wed, 01 Aug 2018 09:00:27 GMT,
server:Hardware management console API web server / 2.0

<No response body>
```

Figure 64. Attach Storage Group to Partition: Response

Stop Partition

The `Stop Partition` operation stops the Partition object designated by *{partition-id}*.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/stop
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the stop request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 270. The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

`Stop Partition` is an orderly process for terminating a partition.

Stopping a partition includes:

- Stopping the execution of (logical) processors associated with the partition.
- Unloading the partition's operating system.
- Freeing non-reserved CPC processor, memory, and adapter resources so that those resources are available for use by other partitions.

After the partition is stopped, the partition is no longer operational.

The operation asynchronously stops the identified partition. When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, as described in "Job status and reason codes" on page 270.

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission. If the user does not have action/task permission to the `Stop Partition` action, a 403 (Forbidden) status code is returned. If the partition is not in a valid state to perform the stop operation, 409 (Conflict) status code is returned. The valid states to perform `Stop Partition` are **"active"**, **"degraded"**, **"paused"**, and **"terminated"** .

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **Stop Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 269.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 126. Stop Partition: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	Stop completed successfully.

Table 126. Stop Partition: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"active"**, **"paused"**, or **"terminated"**) .
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	43	The operation has been canceled.
500 (Server Error)	100	Partition stop failed.
	101	Partition stop job timed out.
	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/stop HTTP/1.1
x-api-session: 44p85d6k4nve3pdaglbdxyohewkhj6hwvnqo9jv0czp226h82p
content-type: application/json
```

Figure 65. Stop Partition: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/eebbef94-d537-11e5-9e2e-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 05:33:10 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/eebbef94-d537-11e5-9e2e-5ef3fcb21ee8"
}
```

Figure 66. Stop Partition: Response

Dump Partition

The `Dump Partition` operation loads a standalone dump program from a designated SCSI device. This operation is not supported when the **"dpm-storage-management"** feature is enabled on the target Partition object.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/scsi-dump
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`dump-load-hba-uri`	String/ URI	Required	The URI of the HBA associated with the partition that provides access to the storage-area network containing the SCSI device from which the dump program is loaded.
`dump-world-wide-port-name`	String (16)	Required	The worldwide port name (WWPN) of the target storage controller that contains the SCSI device from which the dump program is loaded, in hexadecimal.
`dump-logical-unit-number`	String (1-16)	Required	The hexadecimal logical unit number (LUN) that identifies the SCSI device from which the dump program is loaded.
`dump-configuration-selector`	Integer (0-30)	Optional	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system dump program to be loaded. Whether and how this parameter is used to determine boot parameters depends on the dump program and its boot process. Default: 0, which indicates that the dump program's default boot configuration should be used.
`dump-os-specific-parameters`	String	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Default: an empty string.
`dump-record-lba`	String (1-16)	Optional	Specifies the logical block number of the anchor point for locating the operating system dump program on the SCSI disk from which the dump program is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. Default: 0, identifying the first block on the device.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 274. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation loads a standalone dump program into the partition and begins its execution. It does so in such a way that the existing contents of the partition's memory are not overwritten so that the dump program can dump those contents.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 274.

A 404 (Not Found) status code is returned if the object ID *{partition-id}* does not identify a Partition object for which the API user has object-access permission or if the URI in the request body does not designate a resource of an expected type.

If the user does not have action/task permission to the **Dump Partition** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's status is not valid to perform the operation. The valid partition states are **"active"**, **"degraded"**, **"paused"**, or **"terminated"**. A 409 (Conflict) is also returned when there is no storage controller at the WWPN specified, when the storage controller does not have a device with the specified LUN, or if the specified device does not contain a bootable dump program. If the partition has the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **Dump Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 272.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 127. Dump Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate a resource of an expected type

Table 127. Dump Partition: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	12	The operation is not supported when the **"dpm-storage-management"** feature is enabled on the partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 128. Dump Partition: Job status and reason codes		
HTTP error status code	**Reason code**	**Description**
200 (OK)	N/A	Operation completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"active"**, **"paused"**, or **"terminated"**.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	119	The specified device does not contain a bootable dump program.
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/operations/scsi-dump HTTP/1.1
x-api-session: 12a74m6cfuaiiebwuceikrhaqq8bxx1a6kxl6xoihyyfel8x5d
content-type: application/json
content-length: 205
{
   "dump-load-hba-uri":"/api/partitions/d28fc978-d535-11e5-804c-42f2e9cfe851/hbas/bab7e3f8-
      d53a-11e5-a366-42f2e9cfe851",
   "dump-logical-unit-number":"00000",
   "dump-world-wide-port-name":"AFFCB01FF21BCAFA"
}
```

Figure 67. Dump Partition: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 06:01:45 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 68. Dump Partition: Response

Start Dump Program

The `Start Dump Program` operation loads a standalone dump program from a designated external location such as a storage disk.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/start-dump-program
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`dump-program-info`	Object	Required	An object which identifies the location of the dump program and any additional parameters for loading or executing the dump program. The **dump-program-type** field indicates the type dump program and the type of object contained in this field.
`dump-program-type`	String Enum	Required	Indicates the type of dump program and identifies the type of object in **dump-program-info**. Valid values: • **"storage"** - The dump program resides on a storage volume. The **dump-program-info** field contains a storage-volume-dump-program-info object as described in the next table.

The storage-volume-dump-program-info object contains the following fields:

Field name	Type	Rqd/Opt	Description
`storage-volume-uri`	String/ URI	Required	Specifies the storage volume that shall be used to load the dump program. The value has to point to a valid URI of a fulfilled storage volume contained in a storage group attached to the partition.

Field name	Type	Rqd/Opt	Description
`dump-loader-mode`	String Enum	Optional	Specifies the loader mode used when dumping from a FICON storage volume. Values: • **"channel-command-word"** - The default mode available on all SE versions. This value may only be set if **secure-boot** is **false**. • **"list-directed"** - Available when the associated CPC has feature **secure-boot-with-certificates**. This mode supports secure boot for FICON storage volumes and allows additional boot settings. The storage volume must have the correct format to work in **"list-directed"** mode. The value must not be **null** when **storage-volume-uri** points to a FICON storage volume. The value can be set to either **null** or a valid value when **storage-volume-uri** points to a FCP or NVMe storage volume. If the value is not set when **storage-volume-uri** points to a FICON storage volume, the value is initialized automatically to **"channel-command-word"** if **secure-boot** is **false** or **"list-directed"** if **secure-boot** is **true**. [Added by feature **secure-boot-with-certificates**]
`dump-record-location`	boot-record-location object	Optional	Specifies the location of the boot record on the FICON storage volume. If the value is set to **null**, the boot record location is derived from the volume label. The value can be set to either **null** or a valid value when **storage-volume-uri** points to a FCP or NVMe storage volume. Default: null **Note:** Not applicable if **storage-volume-uri** points to a FICON storage volume and **dump-loader-mode** is **"channel-command-word"**. [Added by feature **secure-boot-with-certificates**]

Field name	Type	Rqd/Opt	Description
`dump-configuration`	String Enum	Optional	Specifies how to determine the boot configuration used for booting the dump program. Values: • **"selector"** - Uses the boot configuration specified in **dump-configuration-selector**. • **"automatic"** - The boot loader automatically searches for a boot configuration and uses the first valid boot configuration defined in the operating system. Available only when the associated SE version is 2.16.0 with the suitable MCL bundle, or a later SE version. **Note:** Not applicable if **storage-volume-uri** points to a FICON storage volume and **dump-loader-mode** is **"channel-command-word"**. If the value is not set, it is initialized automatically to **"automatic"** if **dump-configuration-selector** is null or **"selector"** if **dump-configuration-selector** is not null. [Added by feature **secure-boot-with-certificates**]
`dump-configuration-selector`	Integer (0-30)	Optional	Selects the boot configuration to use from among multiple such boot configurations that have been defined by the operating system dump program to be loaded. Whether and how this parameter is used to determine boot parameters depends on the dump program and its boot process. Default: 0 which indicates that the dump program's default boot configuration should be used. **Note:** Not applicable if **storage-volume-uri** points to a FICON storage volume and **"dump-loader-mode"** is set to **"channel-command-word"**. [Updated by feature **secure-boot-with-certificates**]
`secure-boot`	Boolean	Optional	If **true**, the software signature of the dump program will be verified using the certificate(s) assigned to the partition. Starting the dump program will fail if the signatures do not match. Default: **false** [Added by feature **secure-boot-with-certificates**]

Field name	Type	Rqd/Opt	Description
`store-status`	Boolean	Optional	If **true**, the dump stores the current values of the processing unit timer, the clock comparator, the program status word, and the contents of the processor registers in their assigned absolute storage locations. The value is only considered when **storage-volume-uri** points to a FICON storage volume, otherwise it is ignored. Default: **false** [Added by feature **secure-boot-with-certificates**]
`dump-load-parameters`	String (0-8)	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Not valid when **storage-volume-uri** points to a FICON volume. Default : an empty string
`dump-os-specific-parameters`	String (0-256)	Optional	Specifies parameters that are passed unmodified to the loaded operating system dump program as part of the boot process. The way in which these parameters are used depends on the dump program, but in general, these parameters are intended to specify boot-time configuration settings. Default : an empty string
`dump-record-lba`	String (1-16)	Optional	Specifies the logical block number of the anchor point for locating the operating system dump program on the SCSI disk from which the dump program is loaded. The way in which this parameter is used to locate the operating system depends on the operating system and its boot process. Not valid when **storage-volume-uri** points to a FICON volume. Default : 0 (identifying the first block on the device)
`timeout`	Integer (60-600)	Optional	The time in seconds that is waited before the load of the dump program is aborted. Valid only when **storage-volume-uri** points to a FICON volume. Default: 60

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 280. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation loads a standalone dump program into the partition and begins its execution. It does so in a special way that the existing contents of the partition's memory are not overwritten so that the dump program can dump those contents.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 280.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the URI field in the **"dump-program-info"** does not designate a valid object to which the user has object-access permission.

If the user does not have authority to perform the Dump Partition action, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's status is not valid to perform the operation. The valid partition states are **"active"**, **"degraded"**, **"paused"**, or **"terminated"**.

If the partition does not have the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition
- Action/task permission for the **Dump Partition** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 279.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The operation is not supported when the **"dpm-storage-management"** feature is not enabled on the partition.
	2	A URI in the request body does not designate a storage volume in a storage group attached to the partition.
409 (Conflict)	13	The operation is not supported when the **"dpm-storage-management"** feature is not enabled on the partition.
	122	The operation cannot be performed because the **"dump-program-info"** contains a **"storage-volume-uri"** property that does not designate a storage volume that is fulfilled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 129. Start Dump Program: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	Start operation completed successfully.
409 (Conflict)	1	Partition status is not valid to perform the operation (must be **"active"**, **"paused"** or **"terminated"**.)
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states :**"active"**, **"service-required"**, **"degraded"** and **"exceptions"**.)
	119	The specified device does not contain a bootable dump program.
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/a5987806-9af5-11e8-86f5-00106f0ddbc9/operations/start-dump-program HTTP/1.1
 x-api-session: 5o0m316dydfq37m0uo06mqyzg0bf06yq9t8qm4j80kyx18cnc
content-type: application/json
content-length: 222
 {
    "dump-program-info":{
       "dump-load-parameters":"ABCD",
       "storage-volume-uri":"/api/storage-groups/9e5d850c-9fc5-11e8-8c3e-00106f0ddbc9/
         storage-volumes/9e765546-9fc5-11e8-8c3e-00106f0ddbc9"
    },
    "dump-program-type":"storage"
 }
```

Figure 69. Start Dump Program: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/8cbf1be8-a057-11e8-a48e-00106f0d84e1
cache-control: no-cache
date: Wed, 15 Aug 2018 06:50:51 GMT
content-type: application/json;charset=UTF-8
content-length: 60
 {
    "job-uri":"/api/jobs/8cbf1be8-a057-11e8-a48e-00106f0d84e1"
 }
```

Figure 70. Start Dump Program: Response

Perform PSW Restart

The `Perform PSW Restart` operation restarts the first available processor of the Partition object designated by *{partition-id}*.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/psw-restart
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 283. The **job-results** field contains an empty JSON object when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

Information about PSW Restart can be found on the console help system.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, as described in "Job status and reason codes" on page 283.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object for which the API user has object-access permission. If the user does not have action/task permission to the **PSW Restart** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the partition's **status** is not valid to perform this operation. The valid partition states are **"active"**, **"paused"** or **"terminated".**

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission for the **PSW Restart** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 281.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to it.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 130. Perform PSW Restart: Job status and reason codes		
HTTP error status code	**Reason code**	**Description**
200 (OK)	N/A	Operation completed successfully.
409 (Conflict	1	Partition **status** is not valid to perform the operation (must be **"active"**, **"paused"**, or **"terminated"**.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
500 (Server Error)	263	Operation failed.

Example HTTP interaction

```
POST /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/operations/psw-restart
x-api-session: 65aw2jahugn1wop51hsq0c6aldkkx773dz9ulirrvg2z853m4u
content-type: application/json
```

Figure 71. Perform PSW Restart: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
location: /api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8
cache-control: no-cache
date: Wed, 17 Feb 2016 06:01:45 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/ec9c31e8-d53b-11e5-a9b8-5ef3fcb21ee8"
}
```

Figure 72. Perform PSW Restart: Response

Create Virtual Function

The `Create Virtual Function` operation creates a virtual function for the partition with the given identifier.

HTTP method and URI

```
POST /api/partitions/{partition-id}/virtual-functions
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the virtual function's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the virtual function's **description** property.
`adapter-uri`	String/ URI	Required	The URI of the physical Accelerator adapter which will back the new virtual function.
`device-number`	String	Optional	The value to be set as the virtual function's **device-number** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the created virtual function object.

Description

This operation creates a virtual function for the identified partition and then returns the URI of the created object. Upon success, the response includes a **Location** header that provides the URI of the created virtual function object. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing partition or designates a partition for which the API user does not have object-access permission. If the API user doesn't have action/task permission to the **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the Accelerator adapter designated by the request body.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 284.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 131. Create Virtual Function: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The virtual function **name** provided by the user is already in use by another virtual function of the partition, or the provided **device-number** is already in use by another Virtual Function element or by a NIC element of **type "roce"** of the partition.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	The **adapter-uri** in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The partition does not have sufficient resources to perform this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions HTTP/1.1
x-api-session: 63ba4ynuscmvljkablyprvm2ajhr67pa9b9jon5fz4k5jlt7gw
content-type: application/json
content-length: 101
{
   "adapter-uri":"/api/adapters/f36bf9ec-974f-11e5-bfaa-020000000192",
   "name":"New Virtual Function"
}
```

Figure 73. Create Virtual Function: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
   97e6-11e5-9e1e-020000000192
cache-control: no-cache
date: Tue, 01 Dec 2015 04:45:49 GMT
content-type: application/json;charset=UTF-8
content-length: 124
{
   "element-uri":"/api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/
     621c6430-97e6-11e5-9e1e-020000000192"
}
```

Figure 74. Create Virtual Function: Response

Delete Virtual Function

The `Delete Virtual Function` operation removes an existing virtual function designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{virtual-function-id}	String	Element ID of the virtual function object.

Description

This operation deletes the specified virtual function for the identified partition. Upon success, an Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing partition or designates a partition for which the API user does not have object-access permission. If the API user does not have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional state (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 132. Delete Virtual Function: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission
	5	The request URI does not designate an existing virtual function element of an existing partition.
409 (Conflict)	1	Partition **status** is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
   97e6-11e5-9e1e-020000000192 HTTP/1.1
x-api-session: pz3qrhjpvhw10jsfypo8m8qyca10np08wgq8zmdifixd9r629
```

Figure 75. Delete Virtual Function: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 04:52:01 GMT
```

Figure 76. Delete Virtual Function: Response

Get Virtual Function Properties

The `Get Virtual Function Properties` operation retrieves the properties of a single virtual function that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/virtual-functions/{virtual-function-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{virtual-function-id}	String	Element ID of the virtual function for which properties are to be obtained.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the virtual function object.

Description

This operation returns the current properties of the virtual function object that is specified by the request URI. On successful execution, all of the current properties as defined by the "Data model - Virtual Function element object" on page 229 are provided in the response body and HTTP status code 200 (OK) is returned.

A 404 (Not Found) status code is returned if the request URI does not designate an existing virtual function element of an existing partition, or if the API user does not have object-access permission to that partition.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the partition.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 288.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 133. Get Virtual Function Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission.
	5	The request URI does not designate an existing virtual function element of an existing partition.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb/virtual-functions/
  7e89bce4-4479-11e7-9af6-00106f0d81cb HTTP/1.1
x-api-session: 3yg30uqxt7k97dxkei8rxn8mhbvngwka6nq9ftuxk3qhaka69f
```

Figure 77. Get Virtual Function Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 31 May 2017 09:44:55 GMT
content-type: application/json;charset=UTF-8
content-length: 399
{
   "adapter-uri":"/api/adapters/d7077406-f839-11e6-af0f-00106f0d81cb",
   "class":"virtual-function",
   "description":"",
   "device-number":"0001",
   "element-id":"7e89bce4-4479-11e7-9af6-00106f0d81cb",
   "element-uri":"/api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb/virtual-functions/
      7e89bce4-4479-11e7-9af6-00106f0d81cb",
   "fid":null,
   "name":"ttt",
   "parent":"/api/partitions/de4acc6c-361f-11e7-87dd-00106f0d81cb"
}
```

Figure 78. Get Virtual Function Properties: Response

Update Virtual Function Properties

The `Update Virtual Function Properties` operation modifies an existing virtual function that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

`POST /api/partitions/`*`{partition-id}`*`/virtual-functions/`*`{virtual-function-id}`*

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{virtual-function-id}	String	Element ID of the virtual function.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation modifies an existing virtual function specified by the request URI.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing virtual function element of an existing partition, or if the API user does not have object-access permission to that partition. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition.
- Action/task permission to the **Partition Details** task.
- When updating **adapter-uri**, object-access permission to the adapter identified in that URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 134. Update Virtual Function Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	8	The virtual function **name** provided by the user is already in use by another virtual function of the partition, or the provided **device-number** is already in use by another Virtual Function element or by a NIC element of **type "roce"** of the partition.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource or designates a resource for which the API user does not have object-access permission.
	2	The **adapter-uri** in the request body does not designated an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	5	The request URI does not designate an existing virtual function element of an existing partition.
409 (Conflict)	1	Partition status is not valid to perform the operation
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"** or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/virtual-functions/621c6430-
   97e6-11e5-9e1e-020000000192 HTTP/1.1
x-api-session: 1chtesqt42ehx99ayericoofm6gz1gt85wedt8piouotf6doyh
content-type: application/json
content-length: 46
{
   "name":"Virtual Function Name after update"
}
```

Figure 79. Update Virtual Function Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 04:50:36 GMT
<No response body>
```

Figure 80. Update Virtual Function Properties: Response

Create NIC

The `Create NIC` operation creates a NIC for the partition with the given identifier. For a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used for NICs of **type "iqd"** because such NICs are managed via Partition Links of **type hipersockets**. Therefore, creating such a NIC object should be done by sending a corresponding request to the `Modify Partition Link` or `Create Partition Link` operation. It is recommended to use the `Create partition Link` operation as it can create the HiperSockets adapter and the NIC in the same call. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP method and URI

```
POST /api/partitions/{partition-id}/nics
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the NIC's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the NIC's **description** property.
`network-adapter-port-uri`	String/URI	Required, if the adapter **type** is **"roce"** or **"cna"**	The value to be set as the NIC's **network-adapter-port-uri** property. Required if the **type** of the adapter containing the port referenced by the **network-adapter-port-uri** field is **"roce"** or **"cna"**.

Field name	Type	Rqd/Opt	Description
`virtual-switch-uri`	String/ URI	Required, if the adapter **type** is **"osd"** or **"hipersockets"**	The value to be set as the NIC's **virtual-switch-uri** property. Required if the **type** of the adapter containing the port referenced by the **virtual-switch-uri** field is **"osd"** or **"hipersockets"**.
`device-number`	String (4)	Optional	The value to be set as the NIC's **device-number** property. Device number for this NIC. If not provided, a **device-number** is auto-generated.
`ssc-management-nic`	Boolean	Optional	The value to be set as the NIC's **ssc-management-nic** property. Cannot be set to **true** when the partition's **type** is not **"ssc"** or the **type** of the adapter referenced by the **network-adapter-port-uri** is **"roce"** or **"cna"**. Default: false
`function-number`	Integer	Optional	The value to be set as the NIC's **function-number** property.
`function-range`	Integer (1-128)	Optional	The value to set as the NIC's **function-range** property.
`ssc-ip-address-type`	String Enum	Required if **ssc-management-nic** is **true**	The value to be set as the NIC's **ssc-ip-address-type** property. Cannot be set when the partition's **type** is not **"ssc"**.
`ssc-ip-address`	String/ IPv4 Address or String/ IPv6 Address	Required if **ssc-ip-address-type** is **"ipv4"** or **"ipv6"**	The value to be set as the NIC's **ssc-ip-address** property. Cannot be set when the partition's **type** is not **"ssc"**.
`vlan-id`	Integer	Optional	The value to be set as the NIC's **vlan-id** property.
`ssc-mask-prefix`	String	Required if **ssc-ip-address-type** is **"ipv4"** or **"ipv6"**	The value to be set as the NIC's **ssc-mask-prefix**. Cannot be set when the partition's **type** is not **"ssc"**.
`mac-address`	String	Optional	The value to be set as the NIC's **mac-address** property.
`vlan-type`	String Enum	Required if **vlan-id** is provided	The value to be set as the NIC's **vlan-type** property.

Response body contents

On successful completion, the response body contains the URI of the created NIC object.

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the created NIC object.

Description

This operation creates a NIC for the identified partition and then returns the URI of the created object. Upon success, the response includes a **Location** header that provides the URI of the created NIC object. An Inventory Change notification is emitted asynchronously to this operation. The request identifies the NIC's backing adapter by specifying either a network port URI or a virtual switch URI.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state , a 409 (Conflict) status code is returned.

To create a NIC of **type "iqd"** for a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Modify Partition Link` operation is preferred over `Create NIC`, because such NICs are managed via Partition Links. Although `Modify Partition Link` is preferred in this scenario, the `Create NIC` operation can still be used to create new NICs with **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available. In that case the HMC essentially converts the `Create NIC` operation into a corresponding `Modify Partition Link` operation and processes it in a synchronous fashion. The changes in the underlying implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 373 on page 772 for more information. [Updated by feature **dpm-hipersockets-partition-link-management**]

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the owning Partition Link object when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]
- Action/task permission to the **Partition Details** task.
- Action/task permission to the Partition Link Details task when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 292.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 135. Create NIC: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	5	For partitions that are not of **type "ssc"**: • The user specified a valid value for the **vlan-id** property, but did not specify a value for the **vlan-type** property or it was null. • The user did not specify a value for the **vlan-id** property, but specified the value for the **vlan-type** property as **"enforced"**.
	7	The locally administered bit in the value specified for the **mac-address** property is invalid or the **ssc-management-nic** property was set to true, but the **type** of the NIC element is not **"iqd"** or **"osd"**.
	8	For NIC elements of **type "roce"** or **"cna"**, the NIC **name** provided by the user is already in use by another NIC of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "PCI-based device numbers" on page 197 of the partition. For NIC elements of **type "iqd"** or **"osd"**, the NIC **name** provided by the user is already in use by another NIC of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197 or the **mac-address** provided by the user is already in use by another NIC in any of the partitions in the CPC.
	15	**ssc-management-nic** was set to **true**, but no value for **ssc-ip-address-type** was provided. **ssc-ip-address-type** was set to **"ipv4"** or **"ipv6"**, but no value for **ssc-ip-address** or **ssc-mask-prefix** was provided. For partitions that are not of **type "ssc"**: • The user specified a valid value for the **vlan-id** property, but **vlan-type** was null. • The user specified the value for the **vlan-id** property as null, but specified the value of the **vlan-type** property as **"enforced"**.
	18	For partitions of **type "ssc"**: • A non-null value was specified for the **vlan-type** property. • A value for the **mac-address** property was specified for the NIC element of **type "roce"** or **"cna"**. For partitions of other types: • A value for the properties **mac-address**, **vlan-id**, or **vlan-type** was specified for the NIC element of **type "roce"** or **"cna"**.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Table 135. Create NIC: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.
	2	The object ID in the Network Port URI in request body field **network-adapter-port-uri** does not designate an existing Adapter object, or the API user does not have object-access permission to that adapter, or the object ID in the virtual switch URI in the request body field **virtual-switch-uri** does not designate an existing Virtual Switch object, or the API user does not have object-access permission to that virtual switch.
	6	The element ID in the Network Port URI in request body field **network-adapter-port-uri** does not designate an existing adapter port of the adapter.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	8	The property **ssc-management-nic** was set to **true**, but the partition's **type** is not **"ssc"** or a **function-number** for a virtual function was provided, but no physical function was defined yet (on the adapter designated by **network-adapter-port-uri**).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	116	The partition does not have sufficient resources to perform this operation.
	557	The operation failed because it requires the generation of one or more MAC addresses, but the range of available addresses has been exhausted.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

As outlined above, when creating a NIC of **type "iqd"** for a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, a corresponding `Modify Partition Link` operation is performed. Certain HTTP status and reason codes are reported from both operations, therefore it is recommended to consult the response body details in case of failures. [Updated by feature **dpm-hipersockets-partition-link-management**]

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics HTTP/1.1
x-api-session: 35yero3ati5fholesprwebpbn3ktukx59cucm4tt4c86m6n9id
content-type: application/json
content-length: 100
{
   "name":"Nic1",
   "virtual-switch-uri":"/api/virtual-switches/0c797342-9750-11e5-bfaa-020000000192"
}
```

Figure 81. Create NIC: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-11e5-9d1f-
   020000000192
cache-control: no-cache
date: Tue, 01 Dec 2015 05:03:57 GMT
content-type: application/json;charset=UTF-8
content-length: 123
{
   "element-uri":"/api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-
      97e8-11e5-9d1f-020000000192"
}
```

Figure 82. Create NIC: Response

Delete NIC

The `Delete NIC` operation deletes an existing NIC objects that is designated by its element ID and the object ID of the owning partition. For a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used for NICs of **type "iqd"** because such NICs are managed via Partition Links of **type hipersockets**. Therefore, deleting such a NIC object should be done by sending a corresponding request to the `Modify Partition Link` operation. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Description

This operation deletes the specified NIC. Upon success, an Inventory Change notification is emitted asynchronously to this operation.

To delete a NIC of **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Modify Partition Link` operation is preferred over `Delete NIC`, because such NICs are managed via Partition Links. Although `Modify Partition Link` is preferred in this scenario, the `Delete NIC` operation can still be used to delete NICs with **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available. In that case the HMC essentially converts the `Delete NIC` operation into a corresponding `Modify Partition Link` operation and processes it in a synchronous fashion. The changes in the underlying

implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 373 on page 772 for more information. [Updated by feature **dpm-hipersockets-partition-link-management**]

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition. If the API user doesn't have action/task permission to **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the owning Partition Link object when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]
- Action/task permission to the **Partition Details** task.
- Action/task permission to the Partition Link Details task when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 136. Delete NIC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designated an existing Partition object, or the API user does not have object-access permission to it.
	5	The request URI does not designate an existing NIC of an existing partition.

Table 136. Delete NIC: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	8	NIC cannot be deleted as it is set as the partition's boot device.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	100	NIC cannot be deleted as it is the last SSC Management NIC (**ssc-management-nic** is **true**) on an active partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

As outlined above, when deleting a NIC of **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, a corresponding `Modify Partition Link` operation is performed. Certain HTTP status and reason codes are reported from both operations, therefore it is recommended to consult the response body details in case of failures. [Updated by feature **dpm-hipersockets-partition-link-management**]

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
DELETE /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-
   11e5-9d1f-020000000192 HTTP/1.1
x-api-session: 4qhwaoj0lyleh8l57e0z8znsnpphgelqnatcpe5pu5cjk69qeg
```

Figure 83. Delete NIC: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 05:21:30 GMT
<No response body>
```

Figure 84. Delete NIC: Response

Get NIC Properties

The `Get NIC Properties` operation retrieves the properties of a single NIC object that is designated by its element ID and the object ID of the owning partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the NIC object as defined in the "Data model - NIC element object" on page 230. Field names and data types in the JSON object are the same as the property names and data types defined in the "Data model" on page 209.

Description

This operation returns the current properties for the NIC object that is specified by the request URI.

On successful execution, all of the current properties as defined by the "Data model - NIC element object" on page 230 are provided in the response body and HTTP status code 200 (OK) is returned.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object designated by *{partition-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 299.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 137. Get NIC Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The **object-id** in the URI *{partition-id}* does not designated an existing Partition object, or the API user does not have object-access permission to it.
	5	The request URI does not designate an existing NIC of an existing partition.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851/nics/9ff431cc-e92d-11e6-
  9563-42f2e9cfe851 HTTP/1.1
x-api-session: 14ort688m7f2j5yi4tp1eedxo78x7d0ko7qrm7evk5h28p7bzc
```

Figure 85. Get NIC Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Feb 2017 09:11:27 GMT
content-type: application/json;charset=UTF-8
content-length: 551
{
 "class":"nic",
 "description":"",
 "device-number":"0004",
 "element-id":"9ff431cc-e92d-11e6-9563-42f2e9cfe851",
 "element-uri":"/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851/nics/
    9ff431cc-e92d-11e6-9563-42f2e9cfe851",
 "mac-address":"12:34:56:78:9a:bc",
 "name":"ttt",
 "parent":"/api/partitions/fd93be7e-e928-11e6-bcc9-42f2e9cfe851",
 "ssc-ip-address":"2001:0db8:1234:1234::3",
 "ssc-ip-address-type":"ipv6",
 "ssc-management-nic":true,
 "ssc-mask-prefix":"/24",
 "type":"osd",
 "virtual-switch-uri":"/api/virtual-switches/2b0d93e8-b64d-11e6-99ee-42f2e9cfe851",
 "vlan-id":10
 "vlan-type":"enforced"
}
```

Figure 86. Get NIC Properties: Response

Update NIC Properties

The `Update NIC Properties` operation updates the properties of a single NIC object that is designated by its element ID and the object ID of the owning partition. For a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used for NICs of **type "iqd"** because such NICs are managed via Partition Links of type **hipersockets**. Therefore, updating the properties of such a NIC object should be done by sending a corresponding request to the `Modify Partition Link` operation. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP method and URI

```
POST /api/partitions/{partition-id}/nics/{nic-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the Partition object.
{nic-id}	String	Element ID of the NIC.

Request body contents

The request body is expected to contain one or more field names representing writable NIC properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

This operation modifies the NIC properties for the NIC specified by the request URI.

A 404 (Not Found) status code is returned if the request URI does not designate an existing NIC element of an existing partition, or if the API user does not have object-access permission to that partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

To update the properties of a NIC of **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Modify Partition Link` operation is preferred over `Update NIC Properties`, because such NICs are managed via Partition Links. Although `Modify Partition Link` is preferred in this scenario, the `Update NIC Properties` operation can still be used to update the properties of NICs (except for the **virtual-switch-uri**) with **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available. In that case the HMC essentially converts the `Update NIC Properties` operation into a corresponding `Modify Partition Link` operation and processes it in a synchronous fashion. The changes in the underlying implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 373 on page 772for more information.. [Updated by feature **dpm-hipersockets-partition-link-management**]

If the API user doesn't have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the owning Partition Link object when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]
- Action/task permission to the **Partition Details** task.
- Action/task permission to the Partition Link Details task when the NIC element designated by *{nic-id}* is of **type "iqd"** and the Partition exists on a CPC with API feature **dpm-hipersockets-partition-link-management** available. [Updated by feature **dpm-hipersockets-partition-link-management**]
- When updating **network-adapter-port-uri**, object-access permission to the adapter identified in that URI.
- When updating **virtual-switch-uri**, object-access permission to the backing Adapter object of the Virtual Switch object identified in that URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 138. Update NIC Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	5	For partitions that are not of **type "ssc"**: • The user specified a valid value for the property **vlan-id**, but did not specify a value for the property **vlan-type** or was null. • The user did not specify a value for the property **vlan-id** or was null, but specified the value for the property **vlan-type** as **"enforced"**.
	7	The locally administered bit in the value specified for the **mac-address** property is invalid. or the **ssc-management-nic** was set to true, but the **type** of the NIC element is not **"iqd"** or **"osd"**.
	8	For NIC elements of **type "roce"** or **"cna"**, the NIC **name** provided by the user is already in use by another NIC of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "PCI-based device numbers" on page 197 of the partition. For NIC elements of **type "iqd"** or **"osd"**, the NIC **name** provided by the user is already in use by another NIC of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197, or the **mac-address** provided by the user is already in use by another NIC in any of the partitions in the CPC.
	15	**ssc-management-nic** was set to **true**, but no value for **ssc-ip-address-type** was provided. **ssc-ip-address-type** was set to **"ipv4"** or **"ipv6"**, but no value for **ssc-ip-address** or **ssc-mask-prefix** was provided. For partitions that are not of **type "ssc"**: • The user specified a value for both **vlan-id** and **vlan-type** properties, but either or both of the properties values are invalid.
	18	For partitions of **type "ssc"**: • A non-null value was specified for the **vlan-type** property. • A value for the **mac-address** property was specified for the NIC element of **type "roce"**. For partitions of other types: • A value for the properties **mac-address**, **vlan-id**, or **vlan-type** was specified for the NIC element of **type "roce"**.
	19	For NIC elements of **type "iqd"** on a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, the **virtual-switch-uri** property is not writable. [Updated by feature **dpm-hipersockets-partition-link-management**]
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Table 138. Update NIC Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The **object-id** in the URI *{partition-id}* does not designated an existing Partition object, or the API user does not have object-access permission to it.
	2	The object ID in the Network Port URI in request body field **network-adapter-port-uri** does not designate an existing Adapter object, or the API user does not have object-access permission to that adapter, or the object ID in the virtual switch URI in request body field **virtual-switch-uri** does not designate an existing Virtual Switch object, or the API user does not have object-access permission to that virtual switch.
	5	The request URI does not designate an existing NIC of an existing partition.
	6	The element ID in the Network Port URI in request body field **network-adapter-port-uri** does not designate an existing adapter port of the adapter.

Table 138. Update NIC Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out.
	6	The **state** of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	8	The request cannot be processed because the update to **virtual-switch-uri**/**network-adapter-port-uri** attempts to change the switch/port type. **ssc-management-nic** was set to **true**, but the partition's **type** is not **"ssc"**. For partitions that are not of **type "ssc"**: • The user specified a valid value for **vlan-id** but it is not compatible with the current value of **vlan-type**. • The user specified a valid value for **vlan-type** but it is not compatible with the current value of **vlan-id**. • A **function-number** for a virtual function was provided, but no physical function was defined yet (on the adapter designated by **network-adapter-port-uri**).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	18	For partitions of **type "ssc"**: • A non-null value was specified for the **vlan-type** property. • A value for the **mac-address** property was specified for the NIC element of type **"roce"** or **"cna"**. For partitions of other types: • A value for the properties **mac-address**, **vlan-id**, or **vlan-type** was specified for the NIC element of **type "roce"** or **"cna"**.
	557	The operation failed because it requires the generation of one or more MAC addresses, but the range of available addresses has been exhausted.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

As outlined above, when updating the properties of a NIC of **type "iqd"** owned by a Partition on a CPC with API feature **dpm-hipersockets-partition-link-management** available, a corresponding `Modify Partition Link` operation is performed. Certain HTTP status and reason codes are reported from both operations, therefore it is recommended to consult the response body details in case of failures. [Updated by feature **dpm-hipersockets-partition-link-management**]

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/b4c4bf9e-97e0-11e5-9d1f-020000000192/nics/eb6887e4-97e8-11e5-9d1f-
   020000000192 HTTP/1.1
x-api-session: 3nho0b70bohx2ei3wjryvayxgxxnqo7wtqyj3lsifu4kfok2nf
content-type: application/json
content-length: 25
{
   "name":"NicUpdateName"
}
```

Figure 87. Update NIC Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 01 Dec 2015 05:17:57 GMT
<No response body>
```

Figure 88. Update NIC Properties: Response

Increase Crypto Configuration

The `Increase Crypto Configuration` operation can be used to add more elements to an existing (empty or non-empty) crypto configuration.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/increase-crypto-configuration
```

In this request, the URI variable *{partition-id}* is the object ID of the target partition.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`crypto-adapter-uris`	Array of String/ URI	Optional	Array of URIs listing crypto adapters that should be added to the crypto configuration of this partition.
`crypto-domain-configurations`	Array of crypto-domain-configuration objects	Optional	Array of crypto-domain-configuration objects (see Table 113 on page 229) that should be added to the crypto configuration of this partition.

Description

This operation adds the specified adapters and/or domain configurations to the crypto configuration of the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to the **Partition Details** task, 403 (Forbidden) status code is

returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned. A 400 (Bad Request) status code is returned when no lists are provided or both the lists are empty.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition object designated by *{partition-id}*.
- Object-access permission to all crypto adapter objects specified in **crypto-adapter-uris**.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 139. Increase Crypto Configuration: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	101	No lists provided, or both provided lists are empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission to it.
	2	A URI in the request body does not designated an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.

Table 139. Increase Crypto Configuration: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with **"control-usage"**.
	112	The resulting crypto configuration contains at least one domain index with **"control-usage"** which is already configured for **"control-usage"** by another partition.
	113	The list **crypto-adapter-uris** contains an adapter that is already part of the existing crypto configuration.
	114	The list **crypto-domain-configurations** contains a domain index which is already part of the existing crypto configuration.
	119	The resulting crypto configuration is invalid because it does not contain at least one adapter URI.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/increase-crypto-
   configuration HTTP/1.1
x-api-session: 4zzbvb5b1f5i2huscdbya9c3jp5iwoqv1jp5p5qal2vvmpwmil
content-type: application/json
content-length: 166
{
   "crypto-adapter-uris":[
      "/api/adapters/cd8d52b2-d614-11e5-93bf-42f2e9cfe851"
   ],
   "crypto-domain-configurations":[
      {
         "access-mode":"control-usage",
         "domain-index":1
      }
   ]
}
```

Figure 89. Increase Crypto Configuration: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 03 Mar 2016 05:28:51 GMT

<No response body>
```

Figure 90. Increase Crypto Configurations: Response

Change Crypto Domain Configuration

The `Change Crypto Domain Configuration` operation can be used to change the configuration of a single crypto domain that is already configured.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/change-crypto-domain-configuration
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`domain-index`	Integer	Required	Index of the domain to be changed. See Table 113 on page 229.
`access-mode`	String Enum	Required	The new value of the **access-mode** property of the crypto domain configuration identified by the **domain-index**. See Table 113 on page 229.

The structure of crypto domain configuration objects is described in "crypto-configuration object properties" on page 228.

Description

This operation changes the access mode for a crypto domain configuration that is currently included in the crypto configuration of the partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid state, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 140. Change Crypto Domain Configuration: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with **"control-usage"**.
	112	The resulting crypto configuration contains at least one domain index with **"control-usage"** which is already configured for **"control-usage"** by another partition.
	115	The index used in the request is not part of the crypto configuration of the targeted partition.
	125	One or more domains of **type "control-usage"** could not be removed from the crypto configuration because the designated partition is active and the corresponding crypto configuration includes one ore more crypto adapters in state **"online"**. To allow the removal of usage domains, either stop the partition or configure off all crypto adapters in state **"online"** within the operating system running in the partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/change-crypto-
  domain-configuration HTTP/1.1
x-api-session: 4eiaktj22cbpeiya9yxax1af21u0lwyodll9iicrhy6egy3trp
content-type: application/json
content-length: 51
{
   "access-mode":"control-usage",
   "domain-index":1
}
```

Figure 91. Change Crypto Domain Configuration: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 03 Mar 2016 06:01:20 GMT

<No response body>
```

Figure 92. Change Crypto Domain Configuration: Response

Decrease Crypto Configuration

The `Decrease Crypto Configuration` operation can be used to remove some or all elements of an existing (non-empty) crypto configuration.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/decrease-crypto-configuration
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`crypto-adapter-uris`	Array of String/ URI	Optional	Array of URIs listing crypto adapters that should be removed from the crypto configuration of this partition.
`crypto-domain-indexes`	Array of Integer	Optional	Array of integers, listing all crypto domain indexes that should be removed from the crypto configuration of this partition.

Description

This operation removes the specified adapters and/or domain configurations from the crypto configuration of the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. If the API user doesn't have action/task permission to **Partition Details** task 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), or if the CPC is not in a valid

state, a 409 (Conflict) status code is returned. A 400 (Bad Request) status code is returned if no lists were provided or both the lists were empty.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to all crypto adapter objects specified in **crypto-adapter-uris**.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 141. Decrease Crypto Configuration: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	101	No lists provided, or both provided lists are empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission to it.
	2	A URI in request body field **crypto-adapter-uris** does not designated an existing crypto adapter, or the API user does not have object-access permission to it.

Table 141. Decrease Crypto Configuration: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	111	The resulting crypto configuration is invalid because it does not contain at least one domain configuration with **"control-usage"**.
	112	The resulting crypto configuration contains at least one domain index with **"control-usage"** which is already configured for **"control-usage"** by another partition.
	115	The index used in the request is not part of the crypto configuration of the targeted partition.
	117	One or more of the provided adapter URIs are not part of the configuration of the targeted partition.
	119	The resulting crypto configuration is invalid because it contains a domain configuration but does not contain any crypto adapters.
	125	One or more domains of **type "control-usage"** could not be removed from the crypto configuration because the designated partition is active and the corresponding crypto configuration includes one ore more crypto adapters in state **"online"**. To allow the removal of usage domains, either stop the partition or configure off all crypto adapters in state **"online"** within the operating system running in the partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/7eedd6e4-e0fb-11e5-9731-42f2e9cfe851/operations/decrease-crypto-
   configuration HTTP/1.1
x-api-session: 6al0zfk8sivht7i6bd0xeo4byaklto8al0de8fru67wpdn0coz
content-type: application/json
content-length: 109
{
   "crypto-adapter-uris":[
      "/api/adapters/cd8d52b2-d614-11e5-93bf-42f2e9cfe851"
   ],
   "crypto-domain-indexes":[
      1
   ]
}
```

Figure 93. Decrease Crypto Configuration: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 03 Mar 2016 05:32:46 GMT

<No response body>
```

Figure 94. Decrease Crypto Configuration: Response

Zeroize Crypto Domain

The `Zeroize Crypto Domain` operation clears a domain configured for **"control-usage"** on a specific crypto adapter for the given partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/zeroize-crypto-domain
```

In this request, the URI variable *{partition-id}* is the **object-id** of the Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`crypto-adapter-uri`	String/ URI	Required	The canonical URI path of the crypto adapter containing the domain to be zeroized.
`domain-index`	Integer	Required	The index of the domain to be zeroized.

Description

This operation clears the cryptographic keys and non-compliance mode settings within the given domain (which must have an access-mode of **"control-usage"**) of the given crypto adapter.

If the API user does not have action/task permission to the **Zeroize Crypto Domain (API only)** task, a 403 (Forbidden) status code is returned. A 404 (Not found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission or if the **crypto-adapter-uri** field does not identify a crypto adapter object for which the API user has object-access permission.

A 409 (Conflict) status code is returned if the status of the CPC, partition, or adapter is not valid to perform the operation.

Both the adapter identified by **crypto-adapter-uri** and the domain identified by **domain-index** must be part of the crypto configuration of the partition. The identified domain must also be configured for access-mode **"control-usage"**. If any of these preconditions is not met, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

When this operation completes with HTTP status code 204 (No Content), the domain was successfully zeroized on the adapter.

When this operation completes with HTTP status code 409 (Conflict) with reason code 124, the clearing of keys and settings was initiated, but it might or might not have completed. If the adapter status is not **"active"** after receiving HTTP status code 409 (Conflict) with reason code 124, it is recommended to repeat this operation as soon as the status of the adapter becomes **"active"** again.

Although this operation is accepted for all kinds of crypto adapters, it has no effect for adapters of type **"accelerator"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition designated by *{partition-id}*.
- Object-access permission to all crypto adapters specified in the request body.
- Action/task permission to the **Zeroize Crypto Domain (API only)** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 142. Zeroize Crypto Domain: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission for it.
	2	The crypto adapter specified by **crypto-adapter-uri** in the request body does not exist or the API user does not have object-access permission for it.
	4	The partition does not support this operation.

Table 142. Zeroize Crypto Domain: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The status of the partition is not valid to perform the operation (must be one of **"active"**, **"degraded"**, **"paused"**, or **"terminated"**).
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	10	The operation cannot be performed because the SE is in the process of being shut down.
	120	The status of the adapter designated in the request body is not valid to perform the operation (must be **"active"**).
	121	The adapter designated in the request body is not part of the crypto configuration of the partition.
	122	The domain index designated in the request body is not part of the crypto configuration of the partition.
	123	The domain index designated in the request body is configured as **"control"** (must be **"control-usage"**).
	124	The clearing of keys and settings was initiated, but it might or might not have completed.
500 (Server Error)	273	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/5c6ce80a-e402-11e8-8c9d-fa163ebe78b2/operations/zeroize-
   crypto-domain HTTP/1.1
x-api-session: 27daquoih13tq89tkzpivbq28hmghxegydzoa8pxwata3jrpgh
content-type: application/json
content-length: 95
{
   "crypto-adapter-uri":"/api/adapters/196a234a-e3ff-11e8-a662-fa163ebe78b2",
   "domain-index":0
}
```

Figure 95. Zeroize Crypto Domain: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 12 Nov 2018 11:47:29 GMT

<No response body>
```

Figure 96. Zeroize Crypto Domain: Response

Mount ISO Image

The `Mount ISO Image` operation can be used to upload a user provided ISO image. The uploaded ISO image is the image that gets attached to the partition. The contents of the ISO image are specified as binary data in the body of the POST request.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/mount-iso-image
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Query parameters:

Name	Type	Rqd/Opt	Description
`image-name`	String	Required	This field is used as the displayable name of the ISO image contained in the request body. It becomes the value of the **boot-iso-image-name** in the partition properties. The **image-name** must conform to the requirements of the **boot-iso-image-name** property as documented in the "Data model" on page 209.
`ins-file-name`	String	Required	This field is used to set the **boot-iso-ins-file** in the partition properties. The ins-file-name must conform to the requirements of the **boot-iso-ins-file** property as documented in the "Data model" on page 209.

Request body contents

The request body is the binary contents of an ISO image file. A MIME media type of application/octet-stream should be specified as the **content-type** on the request.

Description

This operation uploads an ISO image and associates it to the partition. If this operation is requested when the partition already has an ISO image associated, the newly uploaded image replaces the current one.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object to which the API user has object-access permission. If the API user does not have action permission for the **Partition Details** task, a 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 143. Mount ISO Image: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	The required query parameters have not been specified.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST  /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/mount-iso-image?image-
   name=TestISO&ins-file-name=TestISO.ins HTTP/1.1
x-api-session: pd0nrulei0qsa1mwlpaw7cmq26rnsdcdhtp4w4m9gzse7gybg
```

Figure 97. Mount ISO Image: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
<No response body>
```

Figure 98. Mount ISO Image: Response

Unmount ISO Image

The `Unmount ISO Image` operation unmounts the currently mounted ISO from the identified partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/unmount-iso-image
```

In this request, the URI variable *{partition-id}* is the object ID of the targeted partition.

Description

This operation unmounts an ISO image that is associated to a partition. This operation sets the partition's **boot-iso-image-name** and **boot-iso-ins-file** properties to null.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object to which the API user has object-access permission. If the API user does not have action permission for the **Partition Details** task, a 403 (Forbidden) status code is returned. If the partition is in one of the transitional states (**"starting"** or **"stopping"**), a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the partition does not currently have an ISO image associated with it.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 144. Unmount ISO Image: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The partition with **object-id** *{partition-id}* does not exist, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	118	The partition does not have any image associated with it to perform the operation.
	119	The partition is currently configured to boot from the ISO image. That is, the partition's **boot-device** property is currently set to **"iso-image"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST  /api/partitions/2052747e-52ac-11e5-a8c4-42f2e9cfe851/unmount-iso-image HTTP/1.1
x-api-session: pd0nrulei0qsa1mwlpaw7cmq26rnsdcdhtp4w4m9gzse7gybg
```

Figure 99. Unmount ISO Image: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 17 Feb 2016 05:30:34 GMT
<No response body>
```

Figure 100. Unmount ISO Image: Response

Detach Storage Group from Partition

The `Detach Storage Group from Partition` operation detaches a storage group from a partition specified by the *{partition-id}* portion of the request URI.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/detach-storage-group
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
`storage-group-uri`	String/ URI	Required	The canonical URI of the Storage Group object to be detached from the partition.

Description

The `Detach Storage Group from Partition` operation detaches a storage group from the partition specified by the *{partition-id}* portion of the request URI. The virtual storage resources that were created for this partition will be deleted as part of this operation.

On successful execution, the storage group is detached from the partition and the virtual storage resources created in this storage group specifically for this partition are removed. The operating system in the partition will no longer be able to access the storage volumes defined in the storage group. If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **storage-group-uri** field in the request body must designate an existing Storage Group. If either of these conditions are not met, status code 404 (Not Found) is returned.

In addition, the API user must have action/task permissions to the **Partition Details** task; otherwise, status code 403(Forbidden) is returned.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the **"dpm-storage-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 145. Detach Storage Group from Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the storage group URI in request body field **storage-group-uri** does not designate an existing Storage Group object, or the API user does not have object-access permission to that storage group.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	13	The operation is not supported when the **"dpm-storage-management"** feature is not enabled on the partition.
	117	The **storage-group-uri** does not designate a storage group that is attached to the partition.
	124	The storage group object designated by the *{storage-group-uri}* was busy performing some other operation.
	153	The storage group referenced by the **storage-group-uri** field contains a storage volume that is defined as the boot device for the target partition, which is in one of the active states, and therefore cannot be detached.

Table 145. Detach Storage Group from Partition: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/acab6d72-8107-11e8-9d6f-00106f0d81cb/operations/detach-storage-
   group HTTP/1.1
content-type:application/json,
x-api-session:1zpqlegsp02sdzr2h04uxhc91ou8fdmxvpn0ukxtaouqg6k3e0
content-length:81,
{
    “storage-group-uri”: “/api/storage-groups/519578c6-9569-11e8-a732-00106f0d81cb”
}
```

Figure 101. Detach Storage Group from Partition: Request

```
204 No Content
cache-control:no-cache,
date:Wed, 01 Aug 2018 09:00:28 GMT,
server:Hardware management console API web server / 2.0

<No response body>
```

Figure 102. Detach Storage Group from Partition: Response

Create HBA

The `Create` HBA operation creates an HBA and adds it to the specified partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas
```

In this request, the URI variable *{partition-id}* is the object ID of the partition to which the HBA is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the HBA's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the HBA's **description** property.
`adapter-port-uri`	String/URI	Required	The value to be set as the HBA's **adapter-port-uri** property.

Field name	Type	Rqd/Opt	Description
`device-number`	String (4)	Optional	The value to be set as the HBA's **device-number** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the newly created HBA object.

Description

The `Create` HBA operation creates and adds a new HBA to the partition specified by the *{partition-id}* portion of the request URI.

On successful execution, the **element-uri** field of the response body and the **Location** response header identify the new HBA. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, the **adapter-port-uri** field in the request body must designate an existing Storage Port of an existing adapter, and the API user must have object-access permission to that adapter.

If either of these conditions is not met, status code 404 (Not Found) is returned. In addition, the API user must have action task permission to **Partition Details** task; otherwise, status code 403 (Forbidden) is returned. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or CPC is not in a valid state or if the partition has the **"dpm-storage-management"** feature enabled, 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}* specified in the request URI.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 322, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 146. Create HBA: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The value of **device-number** is not in the range as expected by the "Data model - HBA element object" on page 233.
	8	The HBA **name** provided by the user is already in use by another HBA of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the request URI (*{partition-id}*) does not designated an existing Partition object, or the API user does not have object-access permission to the object.
	2	The object ID in the Storage Port URI in request body field **adapter-port-uri** does not designate an existing Adapter object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy and request timed out
	6	The state of the CPC hosting partition is not valid to perform the operation. It must be in one of these valid states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
	12	The operation is not supported when the **"dpm-storage-management"** feature is enabled on the partition.
	116	The partition does not have sufficient resources to perform this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas HTTP/1.1
x-api-session: p7gzv1s0wo6ops84jpy88eqspkomjrn0u887n2zm8f8l5pcnd
content-type: application/json
content-length: 134
{
   "adapter-port-uri":"/api/adapters/55a89b60-c027-11e5-80b4-020000000192/storage-ports/0",
   "device-number":"1007",
   "name":"MyHba_7"
}
```

Figure 103. Create HBA: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/b6fe29c2-c8b3-11e5-
  bcda-020000000192
cache-control: no-cache
date: Mon, 01 Feb 2016 07:15:09 GMT
content-type: application/json;charset=UTF-8
content-length: 112
{
   "element-uri":"/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/b6fe29c2-c8b3-11e5-
      bcda-020000000192"
}
```

Figure 104. Create HBA: Response

Delete HBA

The `Delete` HBA operation deletes the HBA and removes it from the specified partition.

HTTP method and URI

```
DELETE /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition from which the HBA is to be removed.
{hba-id}	String	Element ID of the HBA to be removed from the partition.

Description

The `Delete` HBA operation deletes the HBA and removes it from the partition specified by the *{partition-id}* portion of the request URI. The HBA to be deleted is identified by the *{hba-id}* variable in the URI.

Upon successfully removing the HBA, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Partition Details** task; otherwise, status code 403 (Forbidden) is returned.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the **"dpm-storage-management"** feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition containing the HBA element specified by the request URI.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 147. Delete HBA: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The element ID in the request URI *{hba-id}* does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the **"dpm-storage-management"** feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting partition is not valid to perform the operation. It must be in one of these valid states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
	151	The HBA cannot be deleted because it is currently identified with the partition's boot device.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
Delete HBA: Request

DELETE /api/partitions/5fca3d9a-c8b2-11e5-97e4-020000000192/hbas/4658f530-
   c8b3-11e5-bcda-020000000192 HTTP/1.1
x-api-session: 1xsytajjb0a8oer7bl2s7e7uarli8qb33vh3kenfgv9ibesj7c
```

Figure 105. Delete HBA: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:55:38 GMT

<No response body>
```

Figure 106. Delete HBA: Response

Update HBA Properties

The `Update HBA Properties` operation updates one or more writable properties of an HBA designated by the element ID and object ID of the hosting partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition that has the HBA.
{hba-id}	String	Element ID of the HBA for which properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates the writable properties of HBA specified by *{hba-id}* as mentioned in the "Data model - HBA element object" on page 233.

On successful execution, the HBA object has been updated with the supplied property values and status code 204 (No Content) is returned without supplying a response body.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The URI path must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned. In addition, the API user must also have action/task permission to the **Partition Details** task as well, otherwise status code 403 (Forbidden) is returned.

The request body is validated against the data model for this object type to ensure that it contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or the CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the **"dpm-storage-management"** feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Partition Details** task.
- Object-access permission to the partition containing the HBA element specified by the request URI.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 148. Update HBA Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The value of **device-number** is not in the range.
	8	The HBA **name** provided by the user is already in use by another HBA of the partition, or the provided **device-number** is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197.
403 (Forbidden)	1	API user does not have the required permission to this operation.
404 (Not Found)	1	The *{partition-id}* in the request URI does not designate an existing Partition object, or the API user does not have object-access permission to it.
	5	The element ID in the request URI *{hba-id}* does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the **"dpm-storage-management"** feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of these valid states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/09aea4b8-c8b3-11e5-a3ad-
   020000000192 HTTP/1.1
x-api-session: 3pwf2qvaioh3fld3vrlwye8zmdq4cckae3s2mtsn5sxu9dv6c0
content-type: application/json
content-length: 44
{
   "device-number":"1007",
   "name":"MyHba_7"
}
```

Figure 107. Update HBA Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT

<No response body>
```

Figure 108. Update HBA Properties: Response

Get HBA Properties

The `Get HBA Properties` operation retrieves the properties of an HBA designated by its element ID and object ID of the hosting partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/hbas/{hba-id}
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition that owns the HBA.
{hba-id}	String	Element ID of the HBA for which properties are to be returned.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the HBA element object as defined in the "Data model - HBA element object" on page 233. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the HBA specified by element ID *{hba-id}*.

On successful execution, all of the current properties as defined by the "Data model - HBA element object" on page 233 are provided in the response body and HTTP status code 200 (OK) is returned.

If the URI path does not designate an existing HBA element of an existing partition or if the API user does not have object-access permission to that partition or if the partition has the **"dpm-storage-management"** feature enabled, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition object in the request URI.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 328.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 149. Get HBA Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action permission to this operation.
404 (Not Found)	1	The object ID in the request URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	5	The element ID in the request URI *{hba-id}* does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the **"dpm-storage-management"** feature is enabled on the partition.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/ea792b18-c8b2-
   11e5-a3ad-020000000192 HTTP/1.1
x-api-session: 2xvikaohvoji3npmjrdx2ezrpi7tll6z6zur1chx864oda1f6u
```

Figure 109. Get HBA Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 10:02:29 GMT
content-type: application/json;charset=UTF-8
content-length: 417
{
   "adapter-port-uri":"/api/adapters/55e59092-c027-11e5-80b4-020000000192/storage-ports/0",
   "class":"hba",
   "description":"Test",
   "device-number":"1003",
   "element-id":"ea792b18-c8b2-11e5-a3ad-020000000192",
   "element-uri":"/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/ea792b18-
      c8b2-11e5-a3ad-020000000192",
   "name":"MyHba_3",
   "parent":"/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192",
   "wwpn":"0000000000000007"
}
```

Figure 110. Get HBA Properties: Response

Reassign Storage Adapter Port

The `Reassign Storage Adapter Port` operation reassigns the backing storage adapter to which the specified HBA is associated to a new backing FCP adapter.

HTTP method and URI

```
POST /api/partitions/{partition-id}/hbas/{hba-id}/operations/reassign-
storage-adapter-port
```

URI variables:

Name	Type	Description
{partition-id}	String	Object ID of the partition that owns the HBA.
{hba-id}	String	Element ID of the HBA for which properties are to be reassigned.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uri`	String/ URI	Required	The value to be set as the HBA's **adapter-port-uri** property.

Description

The `Reassign Storage Adapter Port` operation changes the adapter port associated with specified partition to different adapter port specified in the request body.

On successful execution, status code 204 (No Content) is returned without supplying a response body.

The request URI must designate an existing HBA element of an existing partition and the API user must have object-access permission to that partition, otherwise status code 404 (Not Found) is returned

In addition, the API user must have action/task permission to the **Partition Details** task, otherwise status code 403 (Forbidden) is returned.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or CPC is not in a valid state, 409 (Conflict) status code is returned.

If the partition has the **"dpm-storage-management"** feature enabled, a 404 (Not Found) status code is returned.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition containing the HBA element specified by the request URI.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 150. Reassign Storage Adapter Port: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have required permission for this operation.
404 (Not Found)	1	The object ID in the request URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	2	The object ID in the Storage Port URI in the request body field **adapter-port-uri** does not designate an existing Adapter object, or the API user does not have object-access permission to the object.
	5	The element ID in the request URI *{hba-id}* does not designate an existing HBA of the partition. Note that a partition has no HBA element objects when the **"dpm-storage-management"** feature is enabled on the partition.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with *{partition-id}* was busy and request timed out.
	6	The state of the CPC hosting the partition is not valid to perform the operation. It must be in one of these valid states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192/hbas/9b76b558-c8b2-11e5-97e4-
   020000000192/operations/reassign-storage-adapter-port HTTP/1.1
x-api-session: 4x5d5t0yyqcr52ngjj3zc8y8d1ztaunjkwn8g9bw2uku2iflii
content-type: application/json
content-length: 90
{
   "adapter-port-uri":"/api/adapters/55a89b60-c027-11e5-80b4-020000000192/storage-ports/0"
}
```

Figure 111. Reassign Storage Adapter Port: Request

```
204 No Content
```

Figure 112. Reassign Storage Adapter Port: Response

Send OS Command

The `Send OS Command` operation sends a command to the operating system running in a partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/send-os-cmd
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`is-priority`	Boolean	Optional	An indication of whether this is a priority operating system command. Set to **true** for priority operating system commands or **false** for non-priority operating system commands. The default is **false**.
`operating-system-command-text`	String (1-200)	Required	The text of the operating system command.

Description

This operation sends a command to the operating system running in the Partition targeted by the request URI.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the targeted partition is not available. Some examples are when the Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

On successful execution, the command is sent to the operating system running in the target Partition object and status code 204 (No Content) is returned without supplying a response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required action/task permission to this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designated an existing Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	332	The messages interface is not available.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd/operations/
   send-os-cmd HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 69
{
  "is-priority": false,
  "operating-system-command-text": "help"
}
```

Figure 113. Send OS Command: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT

<No response body>
```

Figure 114. Send OS Command: Response

Open OS Message Channel

The `Open OS Message Channel` operation opens a message channel to the operating system running in a partition for a client of the JMS notification facility (see the "JMS basics" section of Chapter 4 for more information). SSE clients should refer to the Server-Sent Events Stream operations in Chapter 7 for asynchronous OS message support.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/open-os-message-channel
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

An optional request body can be specified as a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`include-refresh-messages`	Boolean	Optional	An indication of whether refresh operating system messages should be sent. Set to **true** to receive refresh messages, or **false** to prevent refresh messages. The default is **true**.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`topic-name`	String (1-128)	The name of the os-message-notification topic.

Description

This operation opens a message channel to the operating system running in the Partition targeted by the request URI. The message channel is implemented as a JMS topic, specifically as an os-message-notification topic. See Chapter 4, “Asynchronous notification,” on page 77 for information on JMS usage on the HMC. The API user can connect to this topic to start the flow of new (and refreshed) operating system messages.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the targeted partition is not available. Some examples are when the Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If an os-message-notification topic already exists for this partition for the current API session, the operation fails.

On successful execution, the message channel is opened and the os-message-notification topic name is returned in the response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*

- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 334.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 151. Open OS Message Channel: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required action/task permission to this operation.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	331	An os-message-notification topic already exists for this partition for the current API session. Use the `Get Notification Topics` operation to determine the topic name.
	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd/operations/open-os-message-channel
  HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 38
{ "include-refresh-messages": false }
```

Figure 115. Open OS Message Channel: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json;charset=UTF-8
{ "topic-name": "mikeuser.1osmsg.cpc1.par1" }
```

Figure 116. Open OS Message Channel: Response

List OS Messages of a Partition

The `List OS Messages of a Partition` operation lists all currently available operating system (OS) messages for a partition.

HTTP method and URI

```
GET /api/partitions/{partition-id}/operations/list-os-messages
```

In this request, the URI variable *{partition-id}* is the object ID of the target partition.

Query parameters:

Name	Type	Rqd/Opt	Description
`begin-sequence-number`	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number less than this are omitted from the results. If not specified, then no such filtering is performed.
`end-sequence-number`	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number greater than this are omitted from the results. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`os-messages`	Array of os-message-info objects	Array of nested os-message-info objects as described in the next table. The array elements are in order of increasing sequence number, unless that number has wrapped back around to 0 in which case the element with sequence number 0 follows the element with the highest sequence number, thus maintaining the temporal order of the messages.

Each nested os-message-info object contains the following fields:

Field name	Type	Description
`sequence-number`	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message. It can be used for filtering purposes.
`message-text`	String	The text of the operating system message.
`message-id`	Long	The message identifier of the operating system message.
`timestamp`	Timestamp	The time when the operating system message was created or -1 if this information is not available from the corresponding operating system.
`sound-alarm`	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (true) or not (false).

Field name	Type	Description
`is-priority`	Boolean	Specifies whether the operating system message is a priority message (true) or not (false). A priority message indicates a critical condition that requires immediate attention.
`is-held`	Boolean	Specifies whether the operating system message is a held message (true) or not (false). A held message is one that requires a response.
`prompt-text`	String	Specifies the prompt text that is associated with this operating system message or null indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.
`os-name`	String	Specifies the name of the operating system that generated this operating system message or null indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the partition in which the operating system is running.

Description

This operation lists the currently available messages from the operating system running in the specified partition. Only a certain amount of OS message data from each partition is preserved by the HMC for retrieval by this operation. If the OS produces more than that amount, the oldest non-held, non-priority OS messages are no longer available. A gap in the sequence numbers indicates a loss of messages. A loss may be due to that space limitation, or it may be due to the deletion of messages by a console user or the OS.

If the request URI does not identify a Partition object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned. In addition to having object-access permission to the partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target partition is not available. Some examples are when the partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If the **begin-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** less than that are omitted from the response. If the **end-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** greater than that are omitted from the response.

If there are no available OS messages for the specified partition or if no OS messages are to be included in the response due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 336.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partitions/05ca8cc6-8d0c-11e7-804a-00106f0dc539/operations/
  list-os-messages HTTP/1.1
x-api-session: 5uj6itqzadjd9rb7o6rjgpf0de5494qop489hv8fdp2k6mfjjf
```

Figure 117. List OS Messages of a Partition: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 31 Aug 2017 02:02:26 GMT
content-type: application/json;charset=UTF-8
content-length: 593
{
   "os-messages":[
      {
         "is-held":false,
         "is-priority":false,
         "message-id":1011,
         "message-text":"Preparing system.\n",
         "os-name":null,
         "prompt-text":"",
         "sequence-number":0,
         "sound-alarm":false,
         "timestamp":-1
      },
      {
         "is-held":false,
         "is-priority":false,
         "message-id":1012,
         "message-text":"Starting system.\n",
         "os-name":null,
         "prompt-text":"",
         "sequence-number":1,
         "sound-alarm":false,
         "timestamp":-1
      },
      {
         "is-held":false,
         "is-priority":false,
         "message-id":1023,
         "message-text":"[   75.476186] systemd-udevd[66]: starting version 208\n",
         "os-name":null,
         "prompt-text":"",
         "sequence-number":2,
         "sound-alarm":false,
         "timestamp":-1
      }
   ]
}
```

Figure 118. List OS Messages of a Partition: Response

Delete Partition OS Message

The `Delete Partition OS Message` operation deletes a single OS message.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/delete-os-message
```

In this request, the URI variable *{partition-id}* is the object ID of the target partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`sequence-number`	Long	Required	The **sequence-number** property of the OS message on the partition to delete.

Description

This operation deletes a specific partition OS message. The OS message to be deleted is uniquely identified by the combination of the *{partition-id}* variable in the URI and the **sequence-number** in the request body.

The URI path must designate an existing partition and the API user must have object-access permission to it; otherwise status code 404 (Not Found) is returned.

The request body must designate an existing OS message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the Operating System Messages task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission for the **Operating System Messages** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission for the **Operating System Messages** task.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to the object.
	336	The **sequence-number** in the request body does not designate an existing OS message on the partition.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/4767c72e-b00d-3a1c-ac89-87f821404a0b/operations/delete-os-message
   HTTP/1.1
x-api-session: 6d1q521ruuym4dhqubv3m77678qbjx7c68wpdpv5v75ut8n1iq
content-type: application/json
content-length: 22
{
   "sequence-number":0
}
```

Figure 119. Delete Partition OS Message: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 15 Nov 2018 17:21:00 GMT

<No response body>
```

Figure 120. Delete Partition OS Message: Response

Get ASCII Console WebSocket URI

The `Get ASCII Console WebSocket URI` operation returns a new WebSocket URI for the ASCII console exposed by the operating system running in this partition.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/get-ascii-console-websocket-uri
```

In this request, the URI variable *{partition-id}* is the object ID of the target Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/Opt	Description
`force-takeover`	Boolean	Optional	Whether the ASCII console session for this partition opened from another HMC should be forcefully closed and taken over by this HMC. Default: False Note: This flag does not affect the ASCII console session for this partition opened from the same HMC.

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body is a JSON object with the following fields:

Field name	Type	Description
`websocket-uri`	String	The WebSocket URI that should be used to connect to the ASCII console exposed for the operating system running in this partition.

Description

The `Get ASCII Console WebSocket URI` operation returns the URI that a WebSocket client should use to connect to the ASCII console of a partition's operating system. The API client should prepend the secure protocol ('wss'), the WebSocket host, and port information to the URI before using a WebSocket client to connect to it. The HMC is the WebSocket host that accepts incoming WebSocket connection requests.

It is possible that another client has already connected to the ASCII console of a partition from the same HMC or from a different HMC. When the client opens a WebSocket connection with the WebSocket URI obtained by specifying the force-takeover flag as **true**, the existing connection from a different HMC is broken and the same ASCII console session is taken over by this client. It is not possible to break or

take over an existing connection to the ASCII console of a partition from the same HMC, even if the force-takeover flag was specified as **true**.

A maximum of 50 WebSocket URIs may be associated with any given API session. If there are already 50 WebSocket URIs associated with the API session, a 409 (Conflict) status code is returned.

Once returned by this operation, a WebSocket URI remains associated with the API session until the corresponding WebSocket is opened and closed.

A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a Partition object to which the API user has object-access permission. If the API user does not have action/task permission for the **Integrated ASCII Console** task, a 403 (Forbidden) status code is returned. If the partition is in any other state than "active" or if there is already an active connection to this partition's ASCII console, a 409 (Conflict) status code is returned unless the active connection is from a different HMC and the force-takeover flag was specified as **true**.

Note: This operation does not create a connection to the partition's ASCII console. A connection does not exist until the client specifically opens a connection using the WebSocket URI returned by this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/Task permission to the **Integrated ASCII Console** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 341.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the **Integrated ASCII Console** task.
404 (Not Found)	1	The Object ID in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	100	There is already a connection to this partition's ASCII console opened from this HMC.
	101	There are already 50 WebSocket URIs associated with this API session.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/dc63eeac-ce0d-11e7-8b57-00106f0dc513/operations/
  get-ascii-console-websocket-uri HTTP/1.1
x-api-session: 4a4f1hj12hldmm26brcpfnydk663gt6gtyxq4iwto26g2r6wq1
content-type: application/json
content-length: 24
{
   "force-takeover":true
}
```

Figure 121. Get ASCII Console WebSocket URI: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 12 Dec 2017 14:50:15 GMT
content-type: application/json;charset=UTF-8
content-length: 85
{
   "websocket-uri":"/api/websock/4a4f1hj12hldmm26brcpfnydk663gt6gtyxq4iwto26g2r6wq1/1"
}
```

Figure 122. Get ASCII Console WebSocket URI: Response

Attach Tape Link to Partition

The `Attach Tape Link to Partition` operation attaches a tape link to a partition.

HTTP method and URI

`POST /api/partitions/`*`{partition-id}`*`/operations/attach-tape-link`

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
`tape-link-uri`	String/ URI	Required	The canonical URI of the Tape Link object to be attached to the partition.

Description

The `Attach Tape Link to Partition` operation attaches the tape link to the partition specified by the *{partition-id}* portion of the request URI.

On successful execution, the tape link gets associated with the partition.

The virtual tape resources are created immediately on the storage ports depending on the status of the unassigned worldwide port names of the tape link. If the status of the worldwide port name is **"validated"**, a previously verified storage port is used to create the virtual tape resource. If the status of the worldwide port name is **"not-validated"**, a virtual tape resource is created without a storage port. In that case, a storage port is assigned to the virtual tape resource later, when DPM successfully verifies connectivity to the defined storage volumes on storage ports.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

This operation enables the operating system in the partition to access the tape library defined in the tape link, through the virtual tape resources that are assigned to storage ports. The operating system may not

be able to access all the volumes defined in the tape link, until the fulfillment state of the tape link is **"complete"**.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **tape-link-uri** field in the request body must designate an existing Tape Link object and the API user must have object-access permission to that tape link.

If either of these conditions are not met, status code 404 (Not Found) is returned. In addition, the API user must have action/task permissions to **Partition Details** task; otherwise, status code 403 (Forbidden) is returned. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the **"dpm-fcp-tape-management"** feature enabled or if the fulfillment state of the tape link is **"incomplete"**, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Object-access permission to the Tape Link object designated by the **tape-link-uri** field.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 152. Attach Tape Link to Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the tape link URI in request body field **tape-link-uri** does not designate an existing Tape Link object, or the API user does not have object-access permission to that tape link.

Table 152. Attach Tape Link to Partition: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	13	The operation is not supported when the **"dpm-fcp-tape-management"** feature is not enabled on the partition.
	118	The tape link specified by **tape-link-uri** is already attached to the partition.
	119	The tape link specified by **tape-link-uri** is already attached to the number of partitions specified in its **max-partitions** property.
	124	The tape link object designated by **tape-link-uri** was busy performing some other operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/f1ff01b6-4f65-11eb-b6ac-fa163ed4d903/operations/attach-tape-link HTTP/1.1
x-api-session: 4b353i6srgy8fv19s5z3izi2rplfh9zu9z0tkxz8l3a2wr1r43
content-type: application/json
content-length: 73
{
   "tape-link-uri":"/api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903"
}
```

Figure 123. Attach Tape Link to Partition: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 14:59:02 GMT

<No response body>
```

Figure 124. Attach Tape Link to Partition: Response

Detach Tape Link from Partition

The `Detach Tape Link from Partition` operation detaches a tape link from a partition specified by the *{partition-id}* portion of the request URI.

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/detach-tape-link
```

In this request, the URI variable *{partition-id}* is the object ID of the partition.

Request body contents

Name	Type	Rqd/Opt	Description
`tape-link-uri`	String/ URI	Required	The canonical URI of the Tape Link object to be detached from the partition.

Description

The `Detach Tape Link from Partition` operation detaches a tape link from the partition specified by the *{partition-id}* portion of the request URI. The virtual tape resources that were created for this partition will be deleted as part of this operation.

On successful execution, the tape link is detached from the partition and the virtual tape resources created in this tape link specifically for this partition are removed. The operating system in the partition will no longer be able to access the tape library targeted by this tape link. If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The URI path must designate an existing Partition object and the API user must have object-access permission to it. In addition, **tape-link-uri** field in the request body must designate an existing Tape Link object. If either of these conditions are not met, status code 404 (Not Found) is returned.

In addition, the API user must have action/task permissions to the Partition Details task; otherwise, status code 403 (Forbidden) is returned.

If the partition is in any of the transitional states (**"starting"** or **"stopping"**) or if the CPC is not in a valid state, 409 (Conflict) status code is returned. If the partition does not have the **"dpm-fcp-tape-management"** feature enabled, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Partition Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 153. Detach Tape Link from Partition: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{partition-id}* does not designate an existing partition object, or the API user does not have object-access permission to the partition.
	2	The object ID in the tape link URI in the request body field **tape-link-uri** does not designate and existing tape link object.
409 (Conflict)	1	Partition status is not valid to perform the operation.
	2	Partition object with ID *{partition-id}* was busy performing some other operation.
	6	The state of the CPC hosting the partition is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.)
	13	The operation is not supported when the **"dpm-fcp-tape-management"** feature is not enabled on the partition.
	117	The **tape-link-uri** does not designate a tape link that is attached to the partition.
	124	The tape link object designated by the *{tape-link-uri}* was busy performing some other operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/f1ff01b6-4f65-11eb-b6ac-fa163ed4d903/operations/detach-tape-link HTTP/1.1
x-api-session: 2y8ezz2la6yh90kcf6c2g576tge9la0q4nzes9tqc6om8fm8jl
content-type: application/json
content-length: 73
{
   "tape-link-uri":"/api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903"
}
```

Figure 125. Detach Tape Link from Partition: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 15:02:12 GMT

<No response body>
```

Figure 126. Detach Tape Link from Partition: Response

Report a Partition Problem

The `Report a Partition Problem` reports and requests service for a problem on a Partition object designated by *{partition-id}*. [Added by feature **report-a-problem**]

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/report-problem
```

In this request, the URI variable *{partition-id}* is the Object ID of the Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`customer-name`	String (0-50)	Optional	Name of the customer. May not contain any double-byte characters or ";". Default: "Unknown"
`customer-phone-number`	String (0-20)	Optional	Phone number of customer. May not contain any double-byte characters or ";". Default: "Unknown"
`problem-description`	String (1-510)	Required	Description of the problem. May not contain any double-byte characters or ";".

Name	Type	Req/Opt	Description
`problem-type`	String Enum	Required	Identifies the type of problem. One of: • **"power"** - Report a problem with the power subsystem. • **"cpc"** - Report a problem with hardware in the processor subsystem. • **"lan"** - Report a problem with the local area network (LAN). • **"software"** - Report a problem with an operating system or other software. • **"io"** -Report a problem with hardware in the input/output (I/O) configuration. • **"health"** - Report the state of the system before applying a maintenance action. • **"other"** - Report a problem that is not adequately described by any other problem type. • **"test"** - Test whether problems can be reported for the selected system.

Description

The `Report a Partition Problem` operation reports a problem for a Partition object and requests service to repair it.

Problems are reported to the support system for the provided system. Reporting a problem sends the information provided in the request and the machine information that identifies the system to the service provider.

Automatic service call reporting must be enabled on the SE associated with the Partition object via the **Remote Service** task to use this operation. If the SE associated with the Partition object does not have automatic service call reporting enabled, a 409 (Conflict) status code is returned.

Upon successful problem creation, a 204 (No Content) status code is returned. If the API user does not have action/task permission to the **Report a Problem** task, a 403 (Forbidden) status code is returned. If the SE associated with the Partition object is unreachable, a 503 (Service Unavailable) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{partition-id}* does not identify a partition object for which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission for the **Report A Problem** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permissions.
404 (Not Found)	1	The object ID in the URI (*{partition-id}*) does not designate an existing Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	600	The operation cannot be performed because the SE does not have automatic service call reporting enabled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/1c3bffa6-a969-11ed-aac1-fa163e898d57/operations/report-problem HTTP/1.1
x-api-session: 5j7jlizop1xfdemin0zzph6vazdqir6sw53hvpgnqlbd2d9bvs
Content-Type: application/json
Content-Length: 138
{
   "customer-name":"Tester",
   "customer-phone-number":"888-888-8888",
   "problem-description":"This is a test IO problem",
   "problem-type":"io"
}
```

Figure 127. Report a Partition Problem: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Fri, 10 Feb 2023 20:15:07 GMT

<No response body>
```

Figure 128. Report a Partition Problem: Response

Get Partition Historical Sustainability Data

Use the `Get Partition Historical Sustainability Data` operation to retrieve partition data on a specific time range. [Added by feature **environmental-metrics**]

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/get-historical-sustainability-data
```

In this request, the URI variable *{partition-id}* is the Object ID of the Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field Name	Type	Rqd/Opt	Description
`range`	String Enum	Optional	Time range for the historical data points. This is the amount of time to be covered by all data points. The possible values are as follows: • **"last-day"** - Last 24 hours. • **"last-week"** - Last 7 days. • **"last-month"** - Last 30 days. • **"last-three-months"** - Last 90 days. • **"last-six-months"** - Last 180 days. • **"last-year"** - Last 365 days. • **"custom"** - From **custom-range-start** to **custom-range-end**. If not specified, the default value is **"last-week"**.
`custom-range-start`	Timestamp	Required if **range** is **"custom"**	Start time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0.
`custom-range-end`	Timestamp	Required if **range** is **"custom"**	End time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than custom-range-start.
`resolution`	String Enum	Optional	Resolution of requested data points. This is the time interval in between data points. For systems where the "environmental-metrics" feature is not available, the minimum resolution is **"one-hour"**. The possible values are as follows: • **"fifteen-minutes"**- 15 minutes. • **"one-hour"**- 60 minutes. • **"one-day"**- 24 hours. • **"one-week"** - 7 days. • **"one-month"** - 30 days. If not specified, the default value is **"one-hour"**.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
`wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the estimated power consumed by the partition, in Watts, at a specific point in time.
`processor-utilization`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing partition processor utilization in percentage, at a specific point in time.

Description

This operation returns an array of available historical data points in a partition designated by *{partition-id}*.

If the **range** field in the request body content is not **"custom"**, **custom-range-start** and **custom-range-end** are ignored and can be omitted from the request. Otherwise, those fields need to be set or HTTP status code 400 (Bad Request) is returned. Additionally, both need to be greater than zero or HTTP status code 400 (Bad Request) will be returned. Finally, **custom-range-end** must be greater than **custom-range-start** or else HTTP status code 400 (Bad Request) is returned. Should the custom range be greater than the existing range of measured data, the operation will complete successfully and return an array with the existing data points.

On successful execution, HTTP status code 200 (OK) is returned with the response body containing properties defined in "Response body contents" on page 351. Should the "environmental-metrics" feature not be available on the HMC, HTTP status code 404.1 (Not Found) is returned. If the same feature is not available on the CPC, HTTP status code 404.4 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*.
- Action/task permission to the **Environmental Dashboard** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 351.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected or its value is not in the permitted range.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body.
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.
	4	The object designated by the request URI does not support the requested operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/partitions/86c261dc-5e86-11ec-bd06-00106f2590a4/operations/get-historical-
sustainability-data HTTP/1.1
x-api-session: 1amjiy7qz72k6ro5b9d19x7qk8loiuttzkt7mzu2imhkkcm1el
Content-Type: application/json
Content-Length: 46
{
   "range":"last-day",
   "resolution":"one-day"
}
```

Figure 129. Get Partition Historical Sustainability Data: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 15 May 2023 16:32:06 GMT
Content-Type: application/json
Content-Length: 115
{
   "processor-utilization":[
      {
         "data":0,
         "timestamp":1684109534120
      }
   ],
   "wattage":[
      {
         "data":115,
         "timestamp":1684109534120
      }
   ]
}
```

Figure 130. Get Partition Historical Sustainability Data: Response

Assign Certificate to Partition

The `Assign Certificate to Partition` operation assigns a certificate of **type "secure-boot"** to the partition with the given identifier. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`POST /api/partitions/`*`{partition-id}`*`/operations/assign-certificate`

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`certificate-uri`	String/ URI	Required	The canonical URI path of the Certificate to be assigned.

Description

This operation assigns the specified certificate of **type "secure-boot"** to the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. 404 (Not Found) is also returned if the **object-id** in the request body **certificate-uri** does not designate an existing Certificate object, or the API user does not have object-access permission to it.

If the API user doesn't have task permissions to the **Assign Secure Boot Certificates** action, 403 (Forbidden) status code is returned. If the partition is busy, or if maximum number of assigned certificates has been reached, or if the certificate is already assigned, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Object-access permission to the certificate.
- Action/task permission to the **Assign Secure Boot Certificates** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 154. Assign Certificate to Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	381	The operation could not be performed because the certificate is expired.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	372	The operation could not be performed because this certificate would exceed limit of 20 certificates per partition.
	373	The operation cannot be performed because the certificate has already been assigned to this partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/98a99980-4a01-11ed-8d1f-fa163e2983be/operations/assign-certificate HTTP/1.1
x-api-session: 3gto4fuax6o28str0hmubbpzkxccr6e7rs3t2dyst1qn54oafh
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/5fb1cd06-49fb-11ed-983e-fa163e61d0f1"
}
```

Figure 131. Assign Certificate to Partition: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Thu, 13 Oct 2022 08:30:41 GMT

<No response body>
```

Figure 132. Assign Certificate to Partition: Response

Unassign Certificate from Partition

The `Unassign Certificate from Partition` operation unassigns a certificate of **type "secure-boot"** from the partition with the given identifier. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/partitions/{partition-id}/operations/unassign-certificate
```

In this request, the URI variable *{partition-id}* is the object ID of the Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`certificate-uri`	String/ URI	Required	The canonical URI path of the Certificate to be unassigned.

Description

This operation unassigns the specified certificate of **type"secure-boot"** from the corresponding partition.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the **object-id** in the URI *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it. 404 (Not Found) is also returned if the **object-id** in the request body **certificate-uri** does not designate an existing Certificate object, or the API user does not have object-access permission to it.

If the API user does not have task permissions to the **Assign Secure Boot Certificates** action, 403 (Forbidden) status code is returned. If the partition is busy, or if the certificate is not assigned to the partition, a 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Partition object designated by *{partition-id}*
- Action/task permission to the **Assign Secure Boot Certificates** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 155. Unassign Certificate from Partition: HTTP status and reason codes

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	370	The operation cannot be performed because the certificate is not assigned.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/partitions/98a99980-4a01-11ed-8d1f-fa163e2983be/operations/unassign-certificate
HTTP/1.1
x-api-session: 3gto4fuax6o28str0hmubbpzkxccr6e7rs3t2dyst1qn54oafh
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/5fb1cd06-49fb-11ed-983e-fa163e61d0f1"
}
```

Figure 133. Unassign Certificate from Partition: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Thu, 13 Oct 2022 08:30:42 GMT

<No response body>
```

Figure 134. Unassign Certificate from Partition: Response

Inventory service data

Information about the partitions managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for partition objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by inventory class, implicitly through a containing category, or by default) that objects of the class **"partition"** are to be included. An entry for a particular partition is included only if the API user has object-access permission to that object as described in the Get Partition Properties operation.

For each partition to be included, the inventory response array includes the following:

- An array entry for the Partition object itself. This entry is a JSON object with the same contents as is specified in the response body contents section for "Get Partition Properties" on page 250. That is, the data provided is the same as would be provided if a `Get Partition Properties` operation were requested targeting this object.
- An array entry for each NIC associated with the partition. For each such NIC, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for "Get NIC Properties" on page 298.
- An array entry for each virtual function associated with the partition. For each such virtual function, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for "Get Virtual Function Properties" on page 287.
- An array entry for each HBA associated with the partition. For each such HBA, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for "Get HBA Properties" on page 328.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single partition. This object would appear as a sequence of array entries in the response array:

```
 {
  "acceptable-status":[
   "active"
  ],
  "access-basic-counter-set":false,
  "access-basic-sampling":false,
  "access-coprocessor-group-set":false,
  "access-crypto-activity-counter-set":false,
  "access-diagnostic-sampling":false,
  "access-extended-counter-set":false,
  "access-global-performance-data":false,
  "access-problem-state-counter-set":false,
  "auto-start":false,
  "autogenerate-partition-id":true,
  "available-features-list":[
   {
    "description":"The DPM storage management approach in which FCP and FICON storage resources
are defined in Storage Groups, which are attached to Partitions.",
    "name":"dpm-storage-management",
    "state":true
   },
   {
    "description":"The DPM enhancement to support FCP tape.",
    "name":"dpm-fcp-tape-management",
    "state":true
   }
  ],
  "boot-configuration-selector":0,
  "boot-device":"test-operating-system",
  "boot-ftp-host":null,
  "boot-ftp-insfile":null,
  "boot-ftp-username":null,
  "boot-iso-image-name":null,
  "boot-iso-ins-file":null,
  "boot-logical-unit-number":"",
  "boot-network-device":null,
  "boot-os-specific-parameters":"",
  "boot-record-lba":"0",
  "boot-removable-media":null,
  "boot-removable-media-type":null,
  "boot-storage-device":null,
  "boot-storage-volume":null,
  "boot-timeout":60,
  "boot-world-wide-port-name":"",
```


Figure 135. Partition object: Sample inventory data - Response (Part 1)

```
  "class":"partition",
  "cp-absolute-processor-capping":false,
  "cp-absolute-processor-capping-value":1.0,
  "cp-processing-weight-capped":false,
  "cp-processors":1,
  "crypto-configuration":null,
  "current-cp-processing-weight":1,
  "current-ifl-processing-weight":1,
  "degraded-adapters":[],
  "description":"",
  "has-unacceptable-status":true,
  "hba-uris":[],
  "ifl-absolute-processor-capping":false,
  "ifl-absolute-processor-capping-value":1.0,
  "ifl-processing-weight-capped":false,
  "ifl-processors":0,
  "initial-cp-processing-weight":100,
  "initial-ifl-processing-weight":100,
  "initial-memory":4096,
  "ipl-load-parameter":"",
  "is-locked":false,
  "maximum-cp-processing-weight":999,
  "maximum-ifl-processing-weight":999,
  "maximum-memory":4096,
  "minimum-cp-processing-weight":1,
  "minimum-ifl-processing-weight":1,
  "name":"partition1",
  "nic-uris":[
   "/api/partitions/56b2ec56-8b18-11eb-a478-fa163ecb75d9/nics/cc85ee26-8d78-11eb-b37a-
fa163ecb75d9"
  ],
```


Figure 136. Partition object: Sample inventory data - Response (Part 2)


```
  "object-id":"56b2ec56-8b18-11eb-a478-fa163ecb75d9",
  "object-uri":"/api/partitions/56b2ec56-8b18-11eb-a478-fa163ecb75d9",
  "os-current-cp-processors":0,
  "os-current-ifl-processors":0,
  "os-current-memory":0,
  "os-name":"",
  "os-type":"",
  "os-version":"",
  "parent":"/api/cpcs/bab22deb-b28a-3821-a675-a0296ca642c4",
  "partition-id":null,
  "permit-aes-key-import-functions":true,
  "permit-cross-partition-commands":false,
  "permit-des-key-import-functions":true,
  "permit-ecc-key-import-functions":true,
  "processor-management-enabled":false,
  "processor-mode":"shared",
  "reserve-resources":false,
  "reserved-memory":0,
  "secure-boot":false,
  "secure-execution":false,
  "short-name":"RRRRR",
  "status":"stopped",
  "storage-group-uris":[
   "/api/storage-groups/96a63db0-8d78-11eb-9c71-fa163ecb75d9"
  ],
  "tape-link-uris":[
   "/api/tape-links/b0fe0786-8b18-11eb-92bd-fa163ecb75d9"
  ],
  "threads-per-processor":1,
  "type":"linux",
  "virtual-function-uris":[]
 }
```


Figure 137. Partition object: Sample inventory data - Response (Part 3)

Adapter object

An Adapter object represents a single adapter for a DPM-enabled CPC. The Adapter object APIs provide access to the set of adapters that are managed by a CPC that is enabled for DPM. APIs exist to query adapters, update selected properties of physical adapters, and create, delete, and update HiperSockets. APIs also exist to query and update properties of the ports of the adapters and to get a list of the partitions for which an adapter is allocated to provide I/O and virtual functions.

Starting with API version 4.1, very limited support for Adapter objects exists for CPCs that are not enabled for DPM (the CPC's **dpm-enabled** property is **false**) to query the complete set of adapters the API user is permitted to access through the APIs and to complete its firmware update. The operations that are supported for non-DPM CPCs are explicitly noted in the operation description. Anything that is not explicitly noted is applicable only for Adapters attached to a CPC that is enabled for DPM.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, including the operational-status-related properties, with the following class-specific specializations:

Table 156. Adapter object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path for an Adapter object is of the form `/api/adapters/{adapter-id}` where `{adapter-id}` is the value of the **object-id** property of the Adapter object.
object-id	—	String (36)	The unique identifier for the adapter instance.
parent	—	String/ URI	The parent of an adapter is conceptually its owning CPC, and so the parent value is the canonical URI path for the CPC.
class	—	String (7)	The class of an Adapter object is **"adapter"**.
name	(w)(pc)	String (1-64)	The display name specified for the adapter. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other configured adapters and created HiperSockets for the CPC.
description	(w)(pc)	String (0-1024)	Arbitrary text providing more descriptive information about the adapter.

Table 156. Adapter object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
status	(sc)	String Enum	The status of the adapter, which must be one of the following values: • **"active"** - The adapter is configured and functioning normally. • **"not-active"** - The adapter is configured as not active. • **"not-detected"** - The previously configured adapter is no longer detected; usually indicates that the card has been physically removed from the system. Not applicable if **type** is **"hipersockets"**. • **"exceptions"** - The adapter is configured but unavailable due to an error affecting the adapter hardware. The adapter is not active. • **"service"** - The adapter is configured but unavailable because it is being serviced.
additional-status	—	String Enum	This property is not supported for adapters and will not be included in its data model.
acceptable-status	—	Array of String Enum	This property is not supported for adapters and will not be included in its data model.

Class specific additional properties

In addition to the properties defined through included schema, this object includes the following additional class-specific properties:

Table 157. Adapter object: class-specific properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
type	(pc)	String Enum	The type of the adapter. Values: • **"crypto"** - Cryptographic adapter. • **"fcp"** - A Fibre Channel attached storage resource. • **"hipersockets"** - A HiperSockets adapter. • **"osd"** - OSA Direct Express. • **"osm"** - OSA-Express for Unified Resource Manager. • **"osc"** - OSA Integrated Console Controller. • **"ose"** - OSE for non-QDIO. • **"roce"** - RDMA over Converged Ethernet. • **"roc2"** - RDMA over Converged Ethernet version 2. • **"zedc"** - zEnterprise Data Compression. • **"fc"** - Fibre Connection attached storage resource. • **"nvme"** - Non-volatile Memory Express. • **"cna"** - Cloud Network Adapter. • **"cl5"** - Long range coupling. • **"cs5"** - Short range coupling. • **"icp"** - Internal Coupling Link. • **"zhyperlink"** - zHyperLink Express. • **"ism"** - Internal Shared Memory. • **"not-configured"** - The adapter is not configured. Currently this applies only to adapters of **adapter-family "ficon"** when the **dpm-enabled** property of the parent CPC is **true**, to all adapters when the **dpm-enabled** property of the parent CPC is **false** and the CPC is not power-on reset complete, and to all adapters of **adapter-family "ficon"** or **"osa"** when the **dpm-enabled** property of the parent CPC is **false** and the adapter has no CHPIDs defined. This value will change when the adapter type is updated through the `Change Adapter Type` operation when the **dpm-enabled** property of the parent CPC is **true**, or when the **dpm-enabled** property of the parent CPC is **false**, and the CPC is power-on reset complete.	All
adapter-id	—	String (3)	ID of the adapter; three character lower-case hex string. This string is the PCHID of the physical adapter or the VCHID of the virtual HiperSockets adapter	All

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
adapter-family	—	String Enum	The family of the card detected for this adapter at the time of its creation. Values: • **"hipersockets"** - A virtual HiperSockets card. • **"osa"** - An OSA card. • **"ficon"** - A FICON card. • **"roce"** - An RDMA over Converged Ethernet card. • **"crypto"** - A cryptographic card. • **"accelerator"** - Adapter card which provides acceleration capability. • **"nvme"** - Non-volatile Memory Express. • **"cna"** - A Cloud Network Adapter card. • **"coupling"** - A coupling card. • **"ism"** - An Internal Shared Memory card. • **"zhyperlink"** - A zHyperLink Express. • **"not-defined"** - The **adapter-family** is not defined. This applies to some adapters if the **dpm-enabled** property of the parent CPC is **false** and the CPC is not powered on.	All

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
detected-card-type	(pc)	String Enum	The type detected for the card with the **adapter-id** of this Adapter object. Values: • **"hipersockets"** - A virtual HiperSockets card • **"osa-express-4s-1gb"** - An OSA Express4S Gigabit Ethernet LX/SX. • **"osa-express-4s-10gb"** - An OSA Express4S Gigabit Ethernet LR/SR. • **"osa-express-4s-1000base-t"** - An OSA Express4S 1000BASE-T Ethernet. • **"osa-express-5s-1gb"** - An OSA Express5S Gigabit Ethernet LX/SX. • **"osa-express-5s-10gb"** - An OSA Express5S Gigabit Ethernet LR/SR. • **"osa-express-5s-1000base-t"** - An OSA Express5S 1000BASE-T Ethernet. • **"osa-express-6s-1gb"** - An OSA Express6S Gigabit Ethernet LX/SX. • **"osa-express-6s-10gb"** - An OSA Express6S Gigabit Ethernet LR/SR. • **"osa-express-6s-1000base-t"** - An OSA Express6s 1000BASE-T Ethernet. • **"osa-express-7s-25gb"** - An OSA Express7S 25 Gigabit Ethernet Short Reach • **"10gbe-roce-express"** - 10 GbE RoCE Express. • **"roce-express-2"** - A 10 GbE RoCE Express2 • **"roce-express-2-25gb"** - A 25 GbE RoCE Express2 • **"crypto-express-5s"** - Crypto Express5S. • **"crypto-express-6s"** - Crypto Express6S. • **"crypto-express-7s"** - Crypto Express7S. • **"crypto-express-8s"** - Crypto Express8S. • **"ficon-express-8"** - FICON Express8 10KM LX (2, 4, 8 Gbps) or FICON Express8 SX (2, 4, 8 Gbps). • **"ficon-express-8s"**- FICON Express8S 10KM LX (2, 4, 8 Gbps) or FICON Express8S SX (2, 4, 8 Gbps). • **"ficon-express-16s"**- FICON Express16S 10KM LX (4, 8, 16 Gbps) or FICON Express16S SX (4, 8, 16 Gbps). • **"ficon-express-16s-plus"** - FICON Express 16S+ LX 10KM (4, 8, 16 Gbps) or FICON Express16S+ SX (4, 8, 16 Gbps).	All

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
			(cont'd) • **"ficon-express-16sa"** - FICON Express 16SA LX 10KM (8, 16 GFC) or FICON Express16SA SX (8, 16 GFC) • **"ficon-express-32s"** - FICON Express32S LX (8, 16, 32 Gbps) or FICON Express32S SX (8, 16, 32 Gbps) • **"fcp-express-32s"** - FCP Express32S LX (16, 32 Gbps) or FCP Express32S SX (16, 32 Gbps) • **"zedc-express"** - zEnterprise Data Compression (FPGA Corsa A5). • **"nvme"** - Non-volatile Memory Express. • **"cloud-network-x5"** - Cloud Network Adapter - X5. • **"cloud-network-x6"** - Cloud Network Adapter - X6. • **"unknown"** - The **detected-card-type** could not be determined.	
card-location	—	String (14 or 20)	Location of the physical I/O card. 14 or 20 characters: "*wwww*-*xxxx*-J.*yy*" for single-port adapters, or "*wwww*-*xxxx*-J.*yy*-J.*zz*" for two-port adapters, where: • *wwww* - ID of the cage in which the card is installed. • *xxxx* - ID of the slot in the cage in which the card is installed. • *yy* - ID of the first jack/port on the card. • *zz* - ID of the second jack/port on the card.	ficon, osa, roce, crypto, accelerator, cna, nvme
port-count	—	Integer	Number of ports on the adapter.	ficon, osa, roce, hipersockets, cna
network-port-uris	—	Array of String/ URI	List of network ports for this adapter. Each element in this array is the canonical URI path of a Network Port object. The number of entries in this list matches the value of the **port-count** property.	osa, roce, hipersockets, cna
storage-port-uris	—	Array of String/ URI	List of storage ports for this adapter. Each element in this array is the canonical URI path of a Storage Port object. The number of entries in this list matches the value of the **port-count** property.	ficon
state	(pc)	String Enum	The current state of the adapter, which must be one of the following values: • **"online"** - The adapter is online. • **"stand-by"** - The adapter has been configured off. • **"reserved"** - The adapter has been configured into service mode. • **"unknown"** - The state of the adapter cannot be determined because of a communications issue.	All

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
maximum-transmission-unit-size	(w)(pc)	Integer Enum	The maximum transmission unit size of the virtual switch. The maximum frame size is implied by this value. Values: • **8** - 8 KB MTU size and 16 KB maximum frame size. • **16** - 16 KB MTU size and 24 KB maximum frame size. • **32** - 32 KB MTU size and 40 KB maximum frame size. • **56** - 56 KB MTU size and 64 KB maximum frame size.	hipersockets
configured-capacity	(pc)	Integer	For FCP adapters (**type** is **"fcp"**), this value is the number of host bus adapters (HBAs) that are configured, but not necessarily allocated, to the partitions assigned to the adapter. For channel-based network adapters (**type** is **"osd"** or **"hipersockets"**), or for channel-based storage adapters of **type "fc"**, this value is the number of subchannels that are configured, but not necessarily allocated, to partitions assigned to the adapter. For Single Root I/O Virtualization (SR-IOV) adapters (**type** is **"roce"**, **"zedc"** or **"cna"**), this value is the number of virtual PCI functions that are configured, but not necessarily allocated, to partitions assigned to the adapter. For adapters with **type** values of **"osm"**, or **"not-configured"**, the value will always be 0. **configured-capacity** may be larger than **allowed-capacity** or **maximum-total-capacity**, which indicates that the adapter is over-committed, and therefore, all partitions that include this adapter in their configuration cannot be active at the same time.	ficon, osa, roce, hipersockets, accelerator, cna
used-capacity	(pc)	Integer	For FCP adapters (**type** is **"fcp"**), this value is the number of host bus adapters (HBAs) that are allocated to partitions assigned to the adapter. For channel-based network adapters (**type** is **"osd"** or **"hipersockets"**), or for channel-based storage adapters of **type "fc"**, this value is the number of subchannels that are allocated to partitions assigned to the adapter. For Single Root I/O Virtualization (SR-IOV) adapters (**type** is **"roce"**, **"zedc"** or **"cna"**), this value is the number of virtual PCI functions that are allocated to partitions assigned to the adapter. For adapters with **type** value **"osm"** or **"not-configured"**, the value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
allowed-capacity	(w)(pc)	Integer	For FCP adapters (**type** is **"fcp"**), this value is the maximum number of host bus adapters (HBAs) that may be allocated to the partitions assigned to the adapter. For FCP adapters, this property is writable and must be an integer from 0 to **maximum-total-capacity**. For channel-based network adapters (**type** is **"osd"** or **"hipersockets"**), or for channel-based storage adapters of **type "fc"**, this value is the maximum number of subchannels that may be allocated to partitions assigned to the adapter. For these adapters, this property is read-only and its value will be automatically defined by the system when the adapter is configured. For Single Root I/O Virtualization (SR-IOV) adapters (**type** is **"roce"**, **"zedc"** or **"cna"**), this value is the maximum number of virtual PCI functions that may be allocated to partitions assigned to the adapter. For these adapters, this property is read-only and its value will be automatically defined by the system when the adapter is configured. For adapters with **type** value **"osm"** or **"not-configured"**, this property is read-only and its value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna
maximum-total-capacity	(pc)	Integer	This is the largest permitted value of **allowed-capacity**. For adapters with **type** value **"osm"** or **"not-configured"**, the value will always be 0.	ficon, osa, roce, hipersockets, accelerator, cna
channel-path-id	(w)(pc)	String (2)	Channel path ID (CHPID) used by the adapter's partition; two character hex string. This value will be **null** when the value of **type** is **"not-configured"**. **channel-path-id** cannot be written if the adapter is configured to any partition.	ficon, osa, hipersockets

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
physical-channel-status	(pc)	String Enum	Status of the physical channel. Values: • **"operating"** - The channel path is operating. Maps to **status** value **"active"**. • **"no-power"** - The power is off for the hardware that supports the channel path, the channel path is not operating. Maps to **status** value **"not-active"**. • **"service"** - The path is in single channel service (SCS) mode and is not in the active I/O configuration. The channel path is not operating. Maps to **status** value **"service"**. • **"stopped"** - The channel path is not operating. Maps to **status** value **"not-active"**. • **"not-defined"** - The channel path is not defined in the active IOCDS. The channel path is not operating. Maps to **status** value **"not-active"**. • **"definition-error"** - The channel path that is specified in the active input/output configuration data set (IOCDS) does not match the characteristics of the installed channel, or the channel type is incompatible with the current storage allocation, or the level of the installed channel hardware does not support the definition in the IOCDS. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"suspended"** - The channel path is suspended. The channel path is not operating. Maps to **status** value **"service"**. • **"check-stopped"** - The channel path is unavailable due to a permanent machine error affecting the channel hardware. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"wrap-block"** - A wrap block is installed on the channel path's channel interface. **Note:** Wrap blocks are used during special diagnostic tests performed on the channel. Wrap blocks must be removed before system initialization to allow the channel to initialize completely. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"permanent-error"** - The channel path is unavailable due to a permanent outboard error. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"initializing"** - The firmware is being loaded into the channel card and then the channel card is starting. Maps to **status** value **"not-active"**.	all

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
			• **"loss-of-signal"** - The channel path detected a link-signal error. The level of the signal on the link is below the value specified for reliable communication. Maps to **status** value **"exceptions"**. • **"loss-of-synchronization"** - The channel path detected a link-signal error. The bit synchronization with the signal was lost. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"not-operational-link"** - The channel path detected a link failure due to a non-operational sequence. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"sequence-time-out"** - The channel path detected a link failure due to a sequence timeout. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"sequence-not-permitted"** - The channel path detected a link failure due to an illegal sequence for a link. The channel path is not operating. Maps to **status** value **"exception"**. • **"terminal-condition"** - The channel path is not available due to an interface-hung condition. This condition can occur after an interface or channel error if the control unit or device fails to disconnect from the interface when requested by the channel. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"offline-signal-received"** - The channel path detected an offline sequence, indicating that the sender is in offline mode and subsequent link-signal errors that are detected by the channel path are not to be reported. For an ES conversion channel, this condition can occur only when the channel is wrongly attached to another channel, switch, or control unit instead of an ESCON Converter. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"fabric-login-sequence-failure"** - This condition means that the channel detected a failure during the Fabric Login procedure. Maps to **status** value **"exceptions"**. • **"port-login-sequence-failure"** - This condition means that the channel detected a failure during the registration procedure. In order for a FICON channel to communicate with devices on a control unit, it must perform a Port Login with that control unit. Maps to **status** value **"exceptions"**. • **"state-change-registration-failure"** - This condition means that the channel detected a failure during the registration procedure. A FICON channel is required to register with the switch to receive state change notification. Maps to **status** value **"exceptions"**.	

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
			• **"invalid-attachment-failure"** - Occurs when the channel determines that it is connected to a switch, but the IOCDS specifies that it should be directly connected to a control unit or the contrary. Maps to **status** value **"exceptions"**. • **"test-mode"** - The channel path is in test mode. The channel path is not operating. Maps to **status** value **"not-active"**. • **"bit-error-threshold-exceeded"** - The number of bit errors the channel path detected while receiving or sending data is more than the threshold set for its bit error counter. The channel path is not operating. Maps to **status** value **"exceptions"**. • **"ifcc-threshold-exceeded"** - The number of interface control checks (IFCCs) the channel path detected is more than the threshold set for its IFCC counter. IFCCs may continue to occur; the error logs will not be created and sent to the Support Element. Maps to **status** value **"exceptions"**. • **"io-supressed"** - The channel path input/output (I/O) suppression is active. I/O suppression prevents the channel subsystem from selecting any device and fetching the first channel command word (CCW) of a channel program. The channel path is not operating. Maps to **status** value **"not-active"**.	
crypto-number	(pc)	Integer	Identifier of the crypto adapter in the range 0-15. Crypto number must be unique to other configured crypto adapters for the CPC.	crypto
crypto-type	(pc)	String Enum	Crypto type. Values: • **"accelerator"** - Crypto Express adapter operating as an Accelerator. • **"cca-coprocessor"** - Crypto Express5S Coprocessor. • **"ep11-coprocessor"** - Crypto Express5S EP11 Coprocessor. • **"not-configured"** - Crypto type is not configured. This applies only to adapters whose parent CPC is not power-on reset complete, and the **dpm-enabled** property of the CPC is **false**. [Updated by feature **adapter-network-information**]	crypto
udx-loaded	(pc)	Boolean	**true** if the configured crypto was UDX-loaded; **false** otherwise. **udx-loaded** cannot be read, and a **null** is returned, if any of the following conditions are true: • The CPC that contains the cryptographic adapter is not started. • The adapter's **status** is **"exceptions"**. • The adapter's **status** is **"not-active"**. • The adapter's **state** is not **"online"**.	crypto

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
tke-commands-enabled	(w)(pc)	Boolean	**true** if the crypto permits TKE commands; **false** otherwise. **tke-command-enabled** cannot be read or written, and a **null** is returned on a read, if any of the following conditions are true: • The CPC that contains the cryptographic adapter is not started. • The adapter's **status** is **"exceptions"**. • The adapter's **status** is **"not-active"**. • The adapter's **state** is not **"online"**.	crypto
ssd-is-installed	(pc)	Boolean	**true** if a solid-state drive (SSD) is installed on the adapter card; **false** otherwise.	nvme
ssd-capacity	(pc)	Float	The capacity in gibibytes (GiB) of the solid-state drive (SSD) installed on the adapter. The capacity is rounded to two decimal places. A **null** is returned if the value of the CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property is **false**.	nvme
ssd-model-number	(pc)	String	The model number of the solid-state drive (SSD) installed on the adapter. A **null** is returned if the value of the CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property is **false**.	nvme
ssd-serial-number	(pc)	String	The serial number of the solid-state drive (SSD) installed on the adapter. A **null** is returned if the value of the CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property is **false**.	nvme
ssd-subsystem-vendor-id	(pc)	Integer	The PCI subsystem vendor identifier of the solid-state drive (SSD) installed on the adapter. A **null** is returned if the value of the CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property is **false**.	nvme
ssd-vendor-id	(pc)	Integer	The PCI vendor identifier of the solid-state drive (SSD) installed on the adapter. A **null** is returned if the value of the CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property is **false**.	nvme

Table 157. Adapter object: class-specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "adapter family" values**
network-ports	(p)	Array of network-port-info objects	An array of network-port-info objects that contain port numbers and MAC addresses, as detailed in Table 158 on page 372. **Note:** This property can be retrieved through the `List Permitted Adapters` operation, but only when the **dpm-enabled** property of the Adapter's parent CPC is **false**. [Added by feature **adapter-network-information**]	osa

Table 158. network-port-info object properties [Added by feature ***adapter-network-information***]			
Name	**Qualifier**	**Type**	**Description**
port-number	—	Integer	The port number.
mac-address	—	String	The MAC address associated with the port. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addresses are valid, e.g. "02:ff:12:34:56:78".

Network Port element object

A Network Port element object defines the index and description associated with a network adapter port.

Table 159. Network Port element object properties			
Name	**Qualifier**	**Type**	**Description**
element-id	—	String (1-2)	Unique ID for the network port within the scope of the containing adapter.
element-uri	—	String/ URI	The canonical URI path for the network port object, of the form `/api/adapters/{adapter-id}/network-ports/{network-port-id}`, where *{adapter-id}* is the **object-id** of the adapter, and *{network-port-id}* is the **element-id** of this network port.
parent	—	String/ URI	The parent of a network port is its owning adapter, so the parent value is the canonical URI path for the adapter.
class	—	String (12)	The class of a network port object is **"network-port"**.
index	—	Integer	The index of the port. Adapters whose **type** is**"osd"** or **"hipersockets"** use a base index of zero while adapters whose **type** is **"roce"** use a base index of one. If adapter **type** is **"osd"** or **"hipersockets"**, values may range from zero to **port-count**. Values may go from one to (**port-count** + 1) if adapter **type** is **"roce"**.
name	—	String (1-64)	The name of the network port, which is currently always of the form "Port #", where # is the **index** of the network port. This form is subject to change in the future.

Table 159. Network Port element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the port.

Storage Port element object

A Storage Port element object defines the index and description associated with a storage adapter port.

Table 160. Storage Port element object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
element-id	—	String (1-2)	Unique ID for the storage port within the scope of the containing adapter.	All
element-uri	—	String/ URI	The canonical URI path for the storage port object, of the form `/api/adapters/{adapter-id}/storage-ports/{storage-port-id}`, where *{adapter-id}* is the **object-id** of the adapter, and *{storage-port-id}* is the **element-id** of this storage port.	All
parent	—	String/ URI	The parent of a storage port is its owning adapter, so the parent value is the canonical URI path for the adapter.	All
class	—	String (12)	The class of a storage port object is **"storage-port"**.	All
index	—	Integer	The index of the port. Adapters whose **type** is **"fcp"** use a base index of zero. If adapter **type** is **"fcp"**, values may range from zero to **port-count**.	All
name	—	String (1-64)	The name of the storage port, which is currently always of the form "Port #", where # is the **index** of the storage port. This form is subject to change in the future.	All
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the port.	All
fabric-id	(pc)	String (16)	The World Wide Name (WWN) of the uplink Fibre Channel switch. When an adapter is not configured into a storage network, **null** is returned.	fcp

Table 160. Storage Port element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
connection-endpoint-class	—	String Enum	The class of the object to which this adapter is connected. Values: • **"storage-subsystem"** – The adapter is directly connected to a storage subsystem. • **"storage-switch"** – The adapter is connected to a storage switch. This value will be **null** when **connection-endpoint-uri** is **null** and non-null when **connection-endpoint-uri** is non-null.	fc

List Adapters of a CPC

The `List Adapters of a CPC` operation lists the adapters managed by the CPC with the given identifier.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/adapters
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern to limit returned objects to those that have a matching **name** property.
`adapter-id`	String	Optional	Filter pattern to limit returned objects to those that have a matching **adapter-id** property.
`adapter-family`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **adapter-family** property. Value must be a valid adapter **adapter-family** property value.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid adapter **type** property.
`status`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **status** property. Value must be a valid adapter **status** property value.

Name	Type	Rqd/Opt	Description
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties (**adapter-family**, **adapter-id**, **name**, **object-uri**, **status**, **type**). This is a list of comma-separated strings where each string is a property name defined in the Adapter object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`adapters`	Array of adapter-info objects	Array of adapter-info objects, described in the next table. Returned array may be empty.

Each nested adapter-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path of the Adapter object.
`name`	String	The **name** property of the Adapter object.
`adapter-id`	String	The **adapter-id** property of the Adapter object.
`adapter-family`	String Enum	The **adapter-family** property of the Adapter object.
`type`	String Enum	The **type** property of the Adapter object.
`status`	String Enum	The **status** property of the Adapter object.

Description

This operation lists the adapters that are managed by the identified CPC. The **object-uri**, **name**, **adapter-id**, **adapter-family**, **type**, and **status** are provided for each.

If the **object-id** *{cpc-id}* does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the CPC identified by *{cpc-id}* is not enabled for DPM, an empty list is returned.

If the **name** or **adapter-id** query parameters are specified, the returned list is limited to those adapters that have a same-named property matching the specified filter pattern. If the **name** or **adapter-id** parameter is omitted, this filtering is not done for the corresponding property.

If the **adapter-family**, **type** or **status** query parameters are specified, the parameter is validated to ensure it is a valid value for the same-named property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those adapters that have a corresponding property matching the specified value. If the **adapter-family**, **type** or **status** parameter is omitted, this filtering is not done for the corresponding property.

An adapter is included in the list only if the API user has object-access permission for that object. If the specified CPC is a manager of an adapter but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If the additional-properties query parameter is specified, the response body is enhanced with the additionally requested properties. The presence and value of each requested property is the same as it would be in the response body of a Get Adapter Properties operation. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the additional-properties query parameter is omitted, only the default properties are included in the response.

If no adapters are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is specified in the request URI.
- Object-access permission to any adapter object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 375.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 161. List Adapters of a CPC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	A CPC with the object ID *{cpc-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29/adapters
x-api-session:3aonctd6ljeg2k4amfb4sz2rb6yo2esgah7dtlpgofor3rzgk1

<no request body>
```

Figure 138. List Adapters of a CPC: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:30:36 GMT",
"content-type":"application/json;charset=UTF-8",
"content-length":"884",
{
 "adapters":[
  {
   "adapter-family":"osa",
   "adapter-id":"18C",
   "name":"OSD 018C Z15B-04",
   "object-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338",
   "status":"active",
   "type":"osd"
  },
  {
   "adapter-family":"crypto",
   "adapter-id":"1C4",
   "name":"Crypto 01C4 Z15B-21",
   "object-uri":"/api/adapters/f7956dc4-c930-11e5-a978-020000000338",
   "status":"not-active",
   "type":"crypto"
  },
  {
   "adapter-family":"ficon",
   "adapter-id":"141",
   "name":"FCP 0141 Z22B-07",
   "object-uri":"/api/adapters/d71902a4-c930-11e5-a978-020000000338",
   "status":"active",
   "type":"fcp"
  }
 ]
}
```

Figure 139. List Adapters of a CPC: Response

List Permitted Adapters

The `List Permitted Adapters` operation lists adapters of both DPM and non-DPM CPCs known to the target console to which the API user has object-access permission. The response body will not contain any Adapter object whose parent is a z15 or earlier CPC that is not enabled for DPM. This operation is also available through the BCPii interface, if the **dpm-enabled** property of the requesting partition's hosting CPC is **false**. [Updated by feature **adapter-network-information**]

HTTP method and URI

```
GET /api/console/operations/list-permitted-adapters
```

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`adapter-id`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **adapter-id** property.
`adapter-family`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **adapter-family** property. Value must be a valid adapter **adapter-family** property value.

Name	Type	Rqd/Opt	Description
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid adapter **type** property.
`status`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **status** property. Value must be a valid adapter **status** property value.
`firmware-update-pending`	Boolean	Optional	Filter string to limit returned objects to those that have a matching **firmware-update-pending** state.
`cpc-name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching **name** property.
`dpm-enabled`	Boolean	Optional	Filter string to limit returned objects to those whose parent CPC has a matching **dpm-enabled** property.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties. This is a list of comma-separated strings where each string is a supported property name defined in the Adapter object's data model. This is supported only when the SE version is 2.16.0 with the suitable MCL bundle or later. The following properties are supported: • **state** • **crypto-type** • **physical-channel-status** • **network-ports** **Note:** If **network-ports** is specified, it is only returned when the **dpm-enabled** value of the adapter's parent CPC is **false**. [Added by feature **adapter-network-information**]

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`adapters`	Array of adapter-info objects	Array of nested adapter-info objects, described in the next table.

Each nested adapter-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the Adapter object.
`name`	String	The **name** property of the Adapter object.
`adapter-id`	String	The **adapter-id** property of the Adapter object.
`adapter-family`	String Enum	The **adapter-family** property of the Adapter object.

Field name	Type	Description
`type`	String Enum	The **type** property of the Adapter object.
`status`	String Enum	The **status** property of the Adapter object.
`firmware-update-pending`	Boolean	The firmware pending state of the adapter. **true** if the adapter has a pending firmware update; **false** otherwise.
`cpc-name`	String	The **name** property of the adapter's parent CPC object.
`cpc-object-uri`	String/ URI	The **object-uri** property of the adapter's parent CPC object.
`se-version`	String	The **se-version** property of the adapter's parent CPC object.
`dpm-enabled`	Boolean	The **dpm-enabled** property of the adapter's parent CPC object.

Description

This operation lists certain Adapter objects to which the API user has object-access permission. Some basic properties are provided for each adapter that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those adapters that have a **name** property matching the specified filter pattern.

If the **adapter-id** query parameter is specified, the returned list is limited to those adapters that have an **adapter-id** property matching the specified filter pattern.

If the **adapter-family** query parameter is specified, the parameter is validated to ensure it is a valid **adapter-family** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those adapters that have an **adapter-family** property matching the specified value.

If the **cpc-name** query parameter is specified, the returned list is limited to those adapters whose parent CPC's **name** property matches the specified filter pattern.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid adapter **type** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those adapters that have a **type** property matching the specified value.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid adapter **status** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those adapters that have a **status** property matching the specified value.

If the **firmware-update-pending** query parameter is specified, the returned list is limited to those adapters that have a firmware pending state matching the specified value.

If the **dpm-enabled** query parameter is specified, the returned list is limited to those adapters whose parent CPC's **dpm-enabled** property matches the specified filter pattern.

If no query parameter is specified, no filtering is performed.

An adapter is included in the list only if the API user has object-access permission to that object. If there is an adapter to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no adapters known to the HMC or if no adapters are to be included in the response due to filtering, parent CPC level, or access permissions, an empty list is provided, and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Adapter objects included in the response body.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object. [Added by feature **adapter-network-information**]

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 378.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 162. List Permitted Adapters: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
403 (Forbidden)	0	The request used in the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object. [Added by feature **bcpii-notifications**]

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/list-permitted-adapters HTTP/1.1
x-api-session: jkl23fdsa7ghjkhgkjlh4hsuaiodg9dsahuif289fsadhh
```

Figure 140. List Permitted Adapters: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 23 Oct 2020 18:00:18 GMT
content-type: application/json;charset=UTF-8
content-length: 686
{
   “adapters” : [
        {
            “adapter-family” : “crypto”,
            “adapter-id” : “100”,
            “cpc-name” : “MYCPC01”,
            “cpc-object-uri” : “/api/cpcs/3efejkdi-3993-3df3-3jdd-dj30003fjdfd”,
            “dpm-enabled” : false,
            “firmware-update-pending” : true,
            “name” : “0100”,
            “object-uri” : “/api/adapters/3940ejdn-3jd9-3099-3jeodj39dn3k”,
            “se-version” : “2.16.0”,
            “status” : “active”,
            “type” : “crypto”
        },
{
   “adapter-family” : “osa”,
   “adapter-id” : “101”,
   “cpc-name” : “MYCPC01”,
   “cpc-object-uri” : “/api/cpcs/3efejkdi-3993-3df3-3jdd-dj30003fjdfd”,
   “dpm-enabled” : false,
   “firmware-update-pending” : false,
   “name” : “0101”,
   “object-uri” : “/api/adapters/3940ejdn-3jd9-3099-3jeudify3928”,
   “se-version” : “2.16.0”,
   “status” : “active”,
   “type” : “osc”
}
]
}
```

Figure 141. List Permitted Adapters: Response

Usage note

The response includes some properties of the parent CPC, regardless of whether the API user has object-access permission to that CPC.

Get Adapter Properties

The `Get Adapter Properties` operation retrieves the properties of a single Adapter object that is designated by its **object-id**.

HTTP method and URI

```
GET /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object for which properties are to be obtained.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Adapter object as defined in the “Data model” on page 360. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the adapter object as defined in the “Data model” on page 360.

If the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose **object-id** is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 381.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	An adapter with object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission to it.
	4	The adapter does not support the operation, because its parent CPC is not enabled for DPM.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/adapters/d71902a4-c930-11e5-a978-020000000338
x-api-session:
   34ccos1y7i88xnu1jbpl9ekpdnar1hwxujg62ots98p3rrvjoy

<no request body>
```

Figure 142. Get Adapter Properties: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:34:11 GMT",
"content-type":"application/json;charset=UTF-8",
"content-length":"670",
{
 "adapter-family":"ficon",
 "adapter-id":"141",
 "allowed-capacity":32,
 "card-location":"Z22B-D207-J.01",
 "channel-path-id":"01",
 "class":"adapter",
 "configured-capacity":1,
 "description":"",
 "detected-card-type":"ficon-express-8",
 "maximum-total-capacity":255,
 "name":"FCP 0141 Z22B-07",
 "object-id":"d71902a4-c930-11e5-a978-020000000338",
 "object-uri":"/api/adapters/d71902a4-c930-11e5-a978-020000000338",
 "parent":"/api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29",
 "physical-channel-status":"operating",
 "port-count":1,
 "state":"online",
 "status":"active",
 "storage-port-uris":[
  "/api/adapters/d71902a4-c930-11e5-a978-020000000338/storage-ports/0"
 ],
 "type":"fcp",
 "used-capacity":1
}
```

Figure 143. Get Adapter Properties: Response

Update Adapter Properties

The `Update Adapter Properties` operation updates one or more of the writable properties of an adapter.

HTTP method and URI

```
POST /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object for which properties are to be updated. For a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used to update the field **"maximum-transmission-unit-size"** for adapters of **type "hipersockets"**, because such adapters are managed via Partition Links. Therefore, updating the **"maximum-transmission-unit-size"** should be done by sending a corresponding request to the `Modify Partition Link` operation. [Updated by feature **dpm-hipersockets-partition-link-management**]

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the "Data model" on page 360. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Fields are only valid if they are supported for an adapter of the targeted type. For instance, an adapter with **type** property **"fcp"** may define a **description** property (all types) but not a **maximum-transmission-unit-size** property (HiperSockets only).

Description

This operation updates an adapter's properties with the values specified.

To update the field **"maximum-transmission-unit-size"** for adapters of **type "hipersockets"** for a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Modify Partition Link` operation is preferred over `Update Adapter Properties`. Although `Modify Partition Link` is preferred in this scenario, the `Update Adapter Properties` operation can still be used. In that case the HMC essentially converts the `Update Adapter Properties` operation for an adapter of **type "hipersockets"** into a corresponding `Modify Partition Link` operation. The changes in the underlying implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 373 on page 772 for more information. [Updated by feature **dpm-hipersockets-partition-link-management**]

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an Adapter object to which the API user has object-access permission. If the adapter **status** is **"definition-error"**, a 409 (Conflict) status code is returned; in this case either a valid card must be installed or the adapter must have its configuration removed and a new configuration added.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the adapter's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 163. Update Adapter Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	An adapter with the **name** specified in the request body already exists on its parent CPC.
	18	An updated property is not valid for an adapter's **adapter-family**.
	420	A crypto adapter for the CPC is already configured with the given **crypto-number**.
403 (Forbidden)	1	API user does not have action permission to the **Adapter Details** task.
404 (Not Found)	1	An adapter with the object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The adapter does not support the operation, because its parent CPC is not enabled for DPM.

Table 163. Update Adapter Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	Adapter **status** is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	Adapter object with the object ID *{adapter-id}* was busy and request timed out.
	420	The given **allowed-capacity** value is not valid because it is less than the current **used-capacity** value.
	421	The **tke-commands-enabled** property cannot be updated when the CPC has not been started.
	422	The **tke-commands-enabled** property cannot be updated when the current **crypto-type** value is **"cca-coprocessor"**.
	423	The **tke-commands-enabled** property cannot be updated when the current **status** value is **"exceptions"**.
	424	The **tke-commands-enabled** property cannot be updated when the current **status** value is **"not-active"**.
	425	The **tke-commands-enabled** property cannot be updated when the current **adapter-state** value is **"online"**.
	429	The **channel-path-id** property cannot be updated when the adapter is configured to any partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/adapters/d71902a4-c930-11e5-a978-020000000338
x-api-session:1nnqtc5mo3yntwilbk96er2jekdekba2nvv3ivphfvmdqym8po
content-type: application/json
content-length: 33
{
  "description":"My FCP adapter"
}
```

Figure 144. Update Adapter Properties: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:35:11 GMT",

<no response body>
```

Figure 145. Update Adapter Properties: Response

Change Crypto Type

The `Change Crypto Type` operation reconfigures a cryptographic adapter to a different crypto type. This operation is only supported for cryptographic adapters.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/operations/change-crypto-type
```

In this request, the URI variable *{adapter-id}* is the object ID of the target cryptographic Adapter object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`crypto-type`	String Enum	Required	The value to be set as the cryptographic adapter's **crypto-type** property. Values: • **"accelerator"** - Crypto Express adapter operating as an Accelerator. • **"cca-coprocessor"** - Crypto Express5S Coprocessor. • **"ep11-coprocessor"** - Crypto Express5S EP11 Coprocessor.
`zeroize`	Boolean	Optional	Specifies whether the cryptographic adapter will be zeroized when it is reconfigured to a **crypto-type** of **"accelerator"**. This field is only valid when **crypto-type** is **"accelerator"**. Default: true

Description

This operation reconfigures a cryptographic adapter to a new crypto type. For a cryptographic adapter that is becoming an accelerator, the adapter may optionally be zeroized as part of the reconfiguration process. A cryptographic adapter must be varied offline before its crypto type can be changed.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify a crypto adapter object to which the API user has object-access permission. If the given crypto-type is not valid, a 400 (Bad Request) status code is returned. If the adapter's **adapter-family** is not **"crypto"**, a 404 (Not Found) status code is returned. If the adapter is online (**state** is **"online"**), a 409 (Conflict) status code is returned. If the CPC that contains the cryptographic adapter has not been started, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the cryptographic adapter's crypto type is changed to the new value specified in the request body.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*

- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 164. Change Crypto Type: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The **crypto-type** value in the request body is not valid.
	15	The **zeroize** value in the request body is not applicable with this **crypto-type**.
403 (Forbidden)	1	The API user does not have action/task permission to the **Adapter Details** task.
404 (Not Found)	1	An adapter with object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission to it.
	4	The operation does not support an adapter of the given **adapter-family** or the Adapter's parent CPC is not enabled for DPM.
409 (Conflict)	1	Adapter **status** is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	Adapter object with ID *{adapter-id}* is not started.
	421	The CPC containing the adapter with object ID *{adapter-id}* is not started.
	426	Adapter **adapter-state** is **"online"**.
	428	The adapter is already configured to be the requested **crypto-type**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/adapters/f7956dc4-c930-11e5-a978-020000000338/operations/change-crypto-type
x-api-session: 4polovirzt2a8qts11nt5ea7mr1it9q1bhez3h3ukh7wvnb8e4
content-type: application/json
content-length: 35
{
  "crypto-type":"ep11-coprocessor"
}
```

Figure 146. Change Crypto Type: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:38:40 GMT",

<no response body>
```

Figure 147. Change Crypto Type: Response

Create Hipersocket

The `Create Hipersocket` operation creates and configures a HiperSockets adapter (**type** is **"hipersockets"**). For a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used to create an adapter of **type "hipersockets"**, because such adapters are managed via Partition Links. Therefore, creating such an adapter should be done by sending a corresponding request to the `Create Partition Link` operation. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/adapters
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the adapter's **name** property. The **name** value must be unique among all other adapters owned by the targeted CPC.
`description`	String (0-1024)	Optional	The value to be set as the adapter's **description** property. Default value: An empty string.
`port-description`	String (0-1024)	Optional	The value to be set as the **description** property of the HiperSocket's single network-port. Default value: An empty string.
`maximum-transmission-unit-size`	Integer Enum	Optional	The value to be set as the adapter's **maximum-transmission-unit-size** property. Default value: 8

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** of the newly created Adapter object.

Description

This operation creates and configures a HiperSocket with the values specified on the identified CPC and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

To create an adapter of **type "hipersockets"** for a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Create Partition Link` operation is preferred over `Create Hipersocket`. Although `Create Partition Link` is preferred in this scenario, the `Create Hipersocket` operation can still be used. In that case the HMC essentially converts the `Create Hipersocket` operation into a corresponding `Create Partition Link` operation. The changes in the underlying implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 361 on page 754 for more information. [Updated by feature **dpm-hipersockets-partition-link-management**]

If the API user does not have action/task permission to **Create HiperSockets Adapter** task, a 403 (Forbidden) status code is returned. If the **object-id** *{cpc-id}* does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the CPC identified by *{cpc-id}* already contains an adapter with the specified **name**, a 400 (Bad Request) status code is returned. If the CPC identified by *{cpc-id}* is not enabled for DPM, a 409 (Conflict) status code is returned. If the CPC identified by *{cpc-id}* already contains the number of HiperSockets identified by its **maximum-hipersockets** property, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given adapter type.

If the request body contents are valid, the HiperSocket is created and its properties are configured to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Create HiperSockets Adapter** task.
- Object-access permission to the CPC whose object ID is *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 388.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 165. Create Hipersocket: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	8	An adapter with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the action/task permission to the **Create HiperSockets Adapter** task.

Table 165. Create Hipersocket: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	A CPC with object ID *{cpc-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The adapter does not support the operation, because its parent CPC is not enabled for DPM.
409 (Conflict)	5	A CPC with object ID *{cpc-id}* is not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	427	A CPC with object ID *{cpc-id}* already has the maximum number of HiperSockets adapters (identified by its **maximum-hipersockets** property) defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29/adapters
x-api-session: 3mwtrej0fkbg71913jr5tvxyl2fd6h35que4x8jrukcl7l1802
content-type: application/json
content-length: 59
{
  "name":"hiper1",
  "description":"My hipersocket adapter"
}
```

Figure 148. Create Hipersocket: Request

```
201 Created
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:41:57 GMT",
"location":"/api/adapters/542b9406-d033-11e5-9f39-020000000338",
"content-type":"application/json;charset=UTF-8",
"content-length":"67",
{
 "object-uri":"/api/adapters/542b9406-d033-11e5-9f39-020000000338"
}
```

Figure 149. Create Hipersocket: Response

Delete Hipersocket

The `Delete Hipersocket` operation deletes a HiperSocket adapter. For a CPC with API feature **dpm-hipersockets-partition-link-management** available, this operation should not be used to delete an adapter of **type "hipersockets"**, because such adapters are managed via Partition Links. Therefore, deleting such an adapter should be done by sending a corresponding request to the `Delete Partition Link` operation. [Updated by feature **dpm-hipersockets-partition-link-management**]

HTTP method and URI

```
DELETE /api/adapters/{adapter-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the adapter to be deleted.

Description

This operation deletes a HiperSocket adapter. An Inventory Change notification is emitted asynchronously to this operation.

To delete an adapter of **type "hipersockets"** for a CPC with API feature **dpm-hipersockets-partition-link-management** available, using the `Delete Partition Link` operation is preferred over `Delete Hipersocket`. Although `Delete Partition Link` is preferred in this scenario, the `Delete Hipersocket` operation can still be used. In that case the HMC essentially converts the `Delete Hipersocket` operation into a corresponding `Delete Partition Link` operation. The changes in the underlying implementation are transparent to the user for successful invocations of the operation. In case of failures, see Table 364 on page 757 for more information [Updated by feature **dpm-hipersockets-partition-link-management**]

If the API user does not have action/task permission to **Delete HiperSocket**, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission. If the adapter is not a HiperSocket (**adapter-family** is not **"hipersockets"**), a 404 (Not Found) is returned. If the adapter is currently configured to any partition, a 409 (Conflict) is returned.

This operation deletes the identified adapter, and removes the HiperSocket adapter from the CPC.

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission to the **Delete HiperSocket** task.
- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 166. Delete Hipersocket: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Delete HiperSocket** task.
404 (Not Found)	1	An adapter with **object-id** *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given **type** or the Adapter's parent CPC is not enabled for DPM.

Table 166. Delete Hipersocket: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	Adapter object with ID *{adapter-id}* was busy and request timed out.
	430	The HiperSocket adapter cannot be deleted because it is currently assigned to one or more partitions.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/adapters/542b9406-d033-11e5-9f39-020000000338
x-api-session:
   62ykfwnk1ds6raibywt03j6ta8g146q06hjneovt2is3lplwl

<no request body>
```

Figure 150. Delete Hipersocket: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:42:51 GMT",

<no response body>
```

Figure 151. Delete Hipersocket: Response

Get Partitions Assigned to Adapter

The `Get Partitions Assigned to Adapter` operation lists the partitions to which the adapter is configured to provide I/O and virtual functions. This operation is not supported for adapters whose **type** is **"osm"**.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/operations/get-partitions-assigned-to-adapter
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object whose partitions should be returned.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern to limit returned Partition objects to those that have a matching **name** property

Name	Type	Rqd/Opt	Description
`status`	String Enum	Optional	Filter string to limit returned Partition objects to those that have a matching **status** property. Value must be a valid partition **status** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`partitions-assigned-to-adapter`	Array of partition-info objects	Array of partition-info objects, described in the next table.

Each nested partition-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path of the Partition object.
`name`	String	The current value of the **name** property of the partition.
`status`	String Enum	The current value of the **status** property of the Partition object.

Description

The `Get Partitions Assigned to Adapter` operation lists the partitions to which the adapter is configured. The object URI, display name, and status information are provided for each.

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid value for the **status** property according to the partition data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have the specified **status** value. If the **status** parameter is omitted, this filtering is not done.

A 404 (Not Found) status code is also returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission. If the adapter's **type** is **"osm"** a 404 (Not Found) status code is returned.

If no partitions are to be included in the results due to filtering (or no partitions exist), an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 393.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 167. Get Partitions Assigned to Adapter: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A **status** query parameter defines an invalid value.
404 (Not Found)	1	An adapter with **object-id** *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	An adapter with **type** of **"osm"** cannot be assigned to any partitions or the Adapter's parent CPC is not enabled for DPM.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/adapters/d71902a4-c930-11e5-a978-020000000338/operations/get-partitions-
   assigned-to-adapter
x-api-session:
   2gr3xkfjj6991vvlwpmnkys8814y68i4anucrudo90ytylfuo5

<no request body>
```

Figure 152. Get Partitions Assigned to Adapter: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:47:42 GMT",
"content-type":"application/json;charset=UTF-8",
"content-length":"140",
{
 "partitions-assigned-to-adapter":[
  {
   "name":"MyPartition",
   "object-uri":"/api/partitions/c0430acc-c9c9-11e5-be4f-020000000338",
   "status":"active"
  }
 ]
}
```

Figure 153. Get Partitions Assigned to Adapter: Response

Get Network Port Properties

The `Get Network Port Properties` operation retrieves the properties of a single Network Port element object.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/network-ports/{network-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{network-port-id}* is the element ID of the Network Port object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the network port object as defined in the "Data model" on page 360. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the network port object as defined in the "Data model" on page 360.

A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the **element-id** *{network-port-id}* does not identify a network port in the adapter. If the adapter **type** is not **"osd"**, **"roce"** or **"hipersockets"**, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 395.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 168. Get Network Port Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	An adapter with object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type or the Adapter's parent CPC is not enabled for DPM.
	5	A network port with **element-id** *{network-port-id}* does not exist for the adapter on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0
x-api-session:
   54dwyb1qutbd5ck8ultlmh752r7lfi3ljbk8pt9814e6j1g1wu

<no request body>
```

Figure 154. Get Network Port Properties: Request

```
200 OK
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:51:27 GMT",
"content-type":"application/json;charset=UTF-8",
"content-length":"229",
{
 "class":"network-port",
 "description":"",
 "element-id":"0",
 "element-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0",
 "index":0,
 "name":"Port 0",
 "parent":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338"
}
```

Figure 155. Get Network Port Properties: Response

Update Network Port Properties

The `Update Network Port Properties` operation updates one or more of the writable properties of a Network Port object. This operation only supports configured adapters whose **type** is **"osd"**, **"osm"**, **"roce"**, or **"hipersockets"**.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/network-ports/{network-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{network-port-id}* is the element ID of the Network Port object.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the "Data model" on page 360. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a network port's properties with the values specified.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the element-id {network-port-id} does not identify a network port in the adapter. If the adapter **type** is not **"osd"**, **"roce"** or **"hipersockets"**, a 404 (Not Found) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported.

If the request body contents are valid, the network port's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 169. Update Network Port Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the **Adapter Details** task.
404 (Not Found)	1	An adapter with the object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type or the Adapter's parent CPC is not enabled for DPM.
	5	A network port with **element-id** *{network-port-id}* does not exist for the adapter on the HMC.
409 (Conflict)	2	Adapter object with ID *{adapter-id}* was busy and request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0
x-api-session: 3y0dr1g0lob607lzurmcwggc2lu4vbjjjiskngz26uno8pa8cz
content-type: application/json
content-length: 30
{
  "description":"My OSA port"
}
```

Figure 156. Update Network Port Properties: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:52:01 GMT",

<no response body>
```

Figure 157. Update Network Port Properties: Response

Get Storage Port Properties

The `Get Storage Port Properties` operation retrieves the properties of a single Storage Port element object.

HTTP method and URI

```
GET /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{storage-port-id}* is the element ID of the Storage Port object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the storage port object as defined in the "Data model" on page 360. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage port object as defined in the "Data model" on page 360.

A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the **element-id** *{storage-port-id}* does not identify a storage port in the adapter. If the adapter **type** is not **"fcp"**, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 398.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 170. Get Storage Port Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Table 170. Get Storage Port Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	An adapter with the object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The adapter does not support the operation, because its parent CPC is not enabled for DPM.
	5	A storage port with **element-id** of *{storage-port-id}* does not exist for the adapter on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0 HTTP/1.1
x-api-session: 46jg2ldhqdx5k27ty3owaiipupfkvmy3bksas8ih3j4wp0exwk
```

Figure 158. Get Storage Port Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 19:07:51 GMT
content-type: application/json;charset=UTF-8
content-length: 361
{
   "class":"storage-port",
   "connection-endpoint-class":"storage-switch",
   "connection-endpoint-uri":"/api/storage-switches/09f4f570-9fe9-11e8-8c0c-fa163e3c2af4",
   "description":"",
   "element-id":"0",
   "element-uri":"/api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
   "index":0,
   "name":"Port 0",
   "parent":"/api/adapters/e7b8811c-9fe7-11e8-bc9a-fa163e3c2af4"
}
```

Figure 159. Get Storage Port Properties: Response

Update Storage Port Properties

The `Update Storage Port Properties` operation updates one or more of the writable properties of a storage port. This operation only supports configured adapters whose **type** is **"fcp"**.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/storage-ports/{storage-port-id}
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object and the URI variable *{storage-port-id}* is the element ID of the Storage Port object.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are

expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage port's properties with the values specified.

If the API user does not have action/task permission to the **Adapter Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{adapter-id}* does not identify an adapter object to which the API user has object-access permission or if the element-id {storage-port-id} does not identify a storage port in the adapter. If the adapter **type** is not **"fcp"**, a 404 (Not Found) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported.

If the **connection-endpoint-uri** field references a storage subsystem when one or more storage fabrics are defined, or if the **connection-endpoint-uri** field references a storage switch or storage subsystem that does not reside in the same CPC as the target storage port, a 409 (Conflict) status code is returned.

If the request body contents are valid, the storage port's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*.
- Action/task permission to the **Adapter Details** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 171. Update Storage Port Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the **Adapter Details** task.
404 (Not Found)	1	An adapter with the object ID *{adapter-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The operation does not support an adapter of the given type or the Adapter's parent CPC is not enabled for DPM.
	5	A storage port with **element-id** of *{storage-port-id}* does not exist for the adapter on the HMC.

Table 171. Update Storage Port Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	Adapter object with the ID *{adapter-id}* was busy and the request timed out
	441	The storage switch or storage subsystem referenced by the **connection-endpoint-uri** field resides in a different CPC than the targeted storage adapter port.
	455	A direct endpoint connection cannot be made to a storage subsystem when storage fabrics are defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/adapters/d71902a4-c930-11e5-a978-020000000338/storage-ports/0
x-api-session: 4sw8s1089cczsejp5vzonbleyrvwsmx6edkqdudskevrcp0bvf
content-type: application/json
content-length: 30
{
  "description":"My FCP port"
}
```

Figure 160. Update Storage Port Properties: Request

```
204 No Content
"server":"zSeries management console API web server / 2.0",
"cache-control":"no-cache",
"date":"Wed, 10 Feb 2016 19:50:05 GMT",

<no response body>
```

Figure 161. Update Storage Port Properties: Response

Change Adapter Type

The `Change Adapter Type` operation reconfigures an adapter from one type to another. Currently, only storage adapters can be reconfigured. Storage adapter instances represent daughter cards on a physical storage card. Current storage cards require both daughter cards to be configured to the same protocol, so changing the configuration of the targeted adapter will also change the configuration of the adapter instance that represents the other daughter card on the same physical adapter. API clients that need to determine the related adapter instance can do so by finding the storage adapter card with a matching first 9 characters (card ID and slot ID) of their **card-location** property values.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/operations/change-adapter-type
```

In this request, the URI variable *{adapter-id}* is the object ID of the Adapter object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`type`	String Enum	Required	The value to be set as the adapter's **type** property. Values: • **"fcp"** - Fibre Channel attached storage resource. • **"fc"** - Fibre Connection attached storage. • **"not-configured"** - The adapter is not configured.

Description

This operation reconfigures a storage adapter to support a new storage protocol.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{adapter-id}* does not identify an Adapter object on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the current value of the adapter's **adapter-family** property is not **"ficon"** or its **detected-card-type** property value is **"ficon-express-32s"**. If the value of the adapter's **status** property is **"exceptions"**, or if the current value of the adapter's **type** property is already set to the value specified in the **type** field in the request body, of if the adapter is configured to any partition, or the adapter's current **type** is **"fc"** and it, or the adapter on the same physical card, is connected to a storage switch, a 409 (Conflict) status code is returned.

If the request body contents are valid, the adapter's **type** property and the **type** property of the adapter with the same card id and slot id, as identified by the first 9 characters of their **card-location** property values, are updated to the value specified in the **type** field in the request body. The two adapters are reconfigured to the new storage protocol.

Property change notifications for the **type** property of both affected adapters are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose object ID is *{adapter-id}*
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 172. Change Adapter Type: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	18	The operation is not allowed for the adapter's **adapter-family** or **detected-card-type**.

Table 172. Change Adapter Type: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	An adapter with object ID *{adapter-id}* does not exist on the HMC.
	4	The adapter does not support the operation, because its parent CPC is not enabled for DPM.
409 (Conflict)	1	Adapter **status** is not valid to perform the operation (does not allow the updating of a specified adapter property).
	2	The adapter object with the **object-id** *{adapter-id}* was busy and the request timed out.
	488	The adapter's **type** property is already set to the value of the **type** field in the request body.
	489	The adapter is configured to at least one partition.
	496	The adapter with object ID *{adapter-id}*, or the adapter on the same physical card, has an endpoint connection to a storage switch.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/adapters/e835bb78-9fe7-11e8-bc9a-fa163e3c2af4/operations/change-adapter-
  type HTTP/1.1
x-api-session: 2fypcb9w0z7kzv9rtq7yfxaypayrdh209nmrrdyv1jetvhxfu0
content-type: application/json
content-length: 15
{
   "type":"fcp"
}
```

Figure 162. Change Adapter Type: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 15 Aug 2018 17:02:33 GMT

<No response body>
```

Figure 163. Change Adapter Type: Response

Update Adapter Firmware

The `Update Adapter Firmware` operation completes the installation and activation of pending firmware updates on a single Adapter object. This operation may be disruptive to workloads if the

adapters have not been configured to operate redundantly. This operation is valid for adapters attached to DPM-enabled CPCs as well as adapters attached to CPCs that are not DPM-enabled.

HTTP method and URI

```
POST /api/adapters/{adapter-id}/operations/update-firmware
```

In this request, the URI variable *{adapter-id}* is the object ID of the target Adapter object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve job status.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 406. The **job-results** field is null when this operation is successful. If the job completion status code is 500 and the reason code is 353, the **job-results** field contains an object with the following field:

Field name	Type	Description
`error-details`	firmware-update-info	Error details contained within a firmware-update-info object, as described in the next table.

Field name	Type	Description
`firmware-update-info`	Array of update-results objects	Array of nested update-results objects as described in the next table.

Configuring an adapter offline may be disruptive to an operating partition if the adapters are not configured to operate redundantly. If the operation encounters an error while configuring an adapter offline or online, the nested update-results object is returned with the name and the URI of partitions that the adapter is assigned to. Each object has the following fields and values:

Field name	Type	Description
`type`	String Enum	Type of failure with one of the following: • configure-off-failed • configure-on-failed
`partition-name`	String	The **name** property of the Partition or Logical Partition to which the adapter is assigned.

Field name	Type	Description
`partition-uri`	String/ URI	The **object-uri** property of the Partition or Logical Partition to which the adapter is assigned.

Description

This operation completes the installation and activation of pending firmware updates. The API client performing the operation needs to be aware that the operation could be disruptive to workloads. The operation will be disruptive to operating system activities if the channels and cryptos are not configured redundantly. As a best practice, it is recommended to have at least two adapters of a given type (CCA, EP11, OSC, etc.) assigned to each partition in order to be concurrent.

The operation will execute asynchronously with the following steps:

- Configure the adapter offline
- Configure the adapter online
- Wait for the adapter to be in operating status.

Note that the target adapter's firmware pending state is cleared if the adapter is successfully configured offline.

This operation will fail in these additional situations:

- The firmware on some other adapter on the same CPC is currently being updated
- An adapter that does not support the operation is targeted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter.
- Object-access permission to all the partitions to which the adapter is assigned
- Action/task permission to the **Manage Adapter Firmware** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 404.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 173. Update Adapter Firmware: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Adapter Firmware** task, or the API user does not have object-access permission to all the partitions to which that Adapter is assigned.
404 (Not Found)	1	The adapter with the object ID *{adapter-id}* does not exist, or the API user does not have object-access permission to it.
	4	The operation does not support an adapter of the given **adapter-family**.

Table 173. Update Adapter Firmware: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The operation cannot be performed because the object designated by the request URI is currently busy performing some other operation.
	350	The operation cannot be performed because firmware on some other Adapter on the same CPC is currently being updated. Only one Adapter can be updated at a time.
	351	The firmware pending state of the adapter is false.
	352	The operation cannot be performed because a reserve is held on the CPC that owns the targeted Adapter.
	353	The operation cannot be performed because the CPC that owns the targeted Adapter is not power-on-reset complete.
500 (Server Error)	280	An IO exception occurred during the scheduling of asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 174. Update Adapter Firmware: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	353	The firmware update of one or more CHPIDs failed. The **error-details** field of the response body contains a **firmware-update-info** object containing additional details.

Example HTTP interaction

```
POST /api/adapters/0af72a18-f251-3f7f-95ff-700406fd9723/operations/update-firmware HTTP/1.1
x-api-session: 3qp3v6nnjdat7eql0ff1y398xtogilf2x5u1nqhasiax1p7oyj
Content-Type: application/json
```

Figure 164. Update Adapter Firmware: Request

```
202
Content-Type: application/json;charset=UTF-8
Content-Length: 60
{
    "job-uri":"/api/jobs/55d1fd34-6e7c-11ec-8243-fa163e45dc1b"
}
```

Figure 165. Update Adapter Firmware: Response

Inventory service data

Information about the Adapters managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Adapter objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "adapter" are to be included. Information for a particular adapter is included only if the API user has object-access permission to that object and the adapter's parent CPC is enabled for DPM.

For each adapter to be included, the inventory response array includes the following:

- An array entry for the Adapter object itself. This entry is a JSON object with the same contents as is specified in the response body contents section for “Get Adapter Properties” on page 381. That is, the data provided is the same as would be provided if a `Get Adapter Properties` operation were requested targeting this object.
- An array entry for each network port associated with the adapter. For each such network port, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for “Get Network Port Properties” on page 394.
- An array entry for each storage port associated with the adapter. For each such storage port, an entry is included that is a JSON object with the same contents as is specified in the response body contents section for “Get Storage Port Properties” on page 398.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single adapter. This object would appear as a sequence of array entries in the response array:

```
{
 "adapter-family":"osa",
 "adapter-id":"18C",
 "allowed-capacity":1920,
 "card-location":"Z15B-D104-J.01",
 "channel-path-id":"10",
 "class":"adapter",
 "configured-capacity":3,
 "description":"My OSA",
 "detected-card-type":"osa-express-5s-10gb",
 "maximum-total-capacity":1920,
 "name":"OSD 018C Z15B-04",
 "network-port-uris":[
  "/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0"
 ],
 "object-id":"e77d39f8-c930-11e5-a978-020000000338",
 "object-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338",
 "parent":"/api/cpcs/87dbe268-0b43-362f-9f80-c79923cc4a29",
 "physical-channel-status":"operating",
 "port-count":1,
 "state":"online",
 "status":"active",
 "type":"osd",
 "used-capacity":3
},
{
 "class":"network-port",
 "description":"My OSA port",
 "element-id":"0",
 "element-uri":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338/network-ports/0",
 "index":0,
 "name":"Port 0",
 "parent":"/api/adapters/e77d39f8-c930-11e5-a978-020000000338"
},
```

Figure 166. Adapter object: Sample inventory data

Virtual Switch object

A Virtual Switch object is a virtualized representation of a CPC's networking adapter and port. Network adapters without a physical port, such as HiperSockets or single port OSAs are virtualized to a single virtual switch. Network adapters with multiple ports are virtualized into multiple virtual switches one for each port. Virtual switches are generated automatically every time a new network adapter is detected and configured. The virtual switch serves as the connection point for network interfaces (VNICs) created by the virtual server administrator.

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This object includes the properties defined in the "Base managed object properties schema" on page 100, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status. The following class-specific specializations apply to the other base managed object properties:

Table 175. Virtual Switch object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path for a Virtual Switch object is of the form `/api/virtual-switches/{vswitch-id}` where `{vswitch-id}` is the value of the **object-id** property of the Virtual Switch object.
object-id	—	String (36)	The unique identifier for the virtual switch instance.
parent	—	String/ URI	The canonical URI path of the CPC object.
class	—	String (14)	The class of a Virtual Switch object is **"virtual-switch"**.
name	(w)(pc)	String (1-64)	The display name of the Virtual Switch object. This name must be unique among all of the CPC's virtual switches, and it must conform to the length and character requirements of the **name** property described in "Base managed object properties schema" on page 100. Default: A string of the form *{PCHID}.{portNumber}.{type}* where *{PCHID}* is the PCHID of the backing adapter, *{portNumber}*is the value of the port property, and *{type}* is the abbreviated type property folded to uppercase. For example, "019F.0.OSD".
description	(w)(pc)	String (0-1024)	The description of the Virtual Switch object, or an empty string. Default: empty string.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 176. Virtual Switch object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
type	—	String Enum	• **"hipersockets"** - a HiperSockets virtual switch. • **"osd"** - OSA Direct Express virtual switch.
backing-adapter-uri	—	String/ URI	The canonical URI path of the backing Adapter object.
port	—	Integer	Physical port identifier associated with the virtual switch. Valid port numbers are 0 and 1. Network adapters that do not have a physical port (HiperSockets) are considered to have a single "virtual" port that is identified as port number 0.
connected-vnic-uris	(p)(pc)(c)	Array of String/ URI	The list of network interfaces (VNICs) connected to this virtual switch. The list is available through the `Get Connected VNICs of a Virtual Switch` operation. Each element in this array is a canonical URI path for a NIC element of a Partition object. This array is initially empty when the virtual switch is created.

List Virtual Switches of a CPC

The `List Virtual Switches of a CPC` operation lists virtual switches that are defined to the CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/virtual-switches
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC for which virtual switches are to be listed.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those objects that have a matching **name** property
`type`	String Enum	Optional	Filter string to limit returned objects to those objects that have a matching **type** property. The value must be a valid **type** property value.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties (**name**, **object-uri**, **type**). This is a list of comma-separated strings where each string is a property name defined in the Virtual Switch object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`virtual-switches`	Array of objects	Array of nested virtual-switch-info objects as described in the next table.

Each nested virtual-switch-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the Virtual Switch object.
`name`	String	The **name** property of the Virtual Switch object
`type`	String Enum	The **type** property of the Virtual Switch object

Description

This operation lists Virtual Switches defined to the CPC. Some basic properties are provided for each virtual switch that is included in the response.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

If the `name` query parameter is specified, the returned list is limited to those virtual switches that have a **name** property matching the specified filter pattern. If the `name` parameter is omitted, no such filtering is performed.

If the `type` query parameter is specified, the parameter is validated to ensure that it is a valid virtual switch **type** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those virtual switches that have a **type** property matching the specified value. If the `type` parameter is omitted, no such filtering is performed.

If the **additional-properties** query parameter is specified, the response body is enhanced with the additionally requested properties. The presence and value of each requested property is the same as it would be in the response body of a `Get Virtual Switch Properties` operation. That is, it may be omitted or contain a special value, such as null, `-1`, or an empty string, if a prerequisite condition is not met. If the **additional-properties** query parameter is omitted, only the default properties are included in the response.

A virtual switch is included in the list only if the API user has object-access permission to the backing adapter of that virtual switch. If there is a virtual switch to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If the CPC is not in DPM mode, or there are no virtual switches defined to the CPC, or no virtual switches are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object identified in the request URI
- Object-access permission to the backing adapter of the Virtual Switch objects to be included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 409.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or it designates a resource for which the API user does not have object-access permission.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef/virtual-switches
x-api-session: 1jgbldxyy2inf0p7aaj9a8p87j7awxsr1mstmfnw07hvoaz8da
```

Figure 167. List Virtual Switches of a CPC: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control:no-cache
date: Tue, 31 Mar 2015 06:20:54 GMT
content-type: application/json;charset=UTF-8
content-length: 255
{
  "virtual-switches":[
  {
   "name":"PrimeIQDVSwitch1",
   "object-uri":"/api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058",
   "type":"hipersockets"
  },
  {
   "name":"5F1.P0.OSD",
   "object-uri":"/api/virtual-switches/dff0b71c-d491-11e4-a555-020000003058",
   "type":"osd"
  }
 ]
}
```

Figure 168. List Virtual Switches of a CPC: Response

List Permitted Virtual Switches

The `List Virtual Switches of a CPC` operation lists virtual switches that are defined to the CPC. [Added by feature dpm-hipersockets-partition-link-management]

HTTP method and URI

```
GET /api/console/operations/list-permitted-virtual-switches
```

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those objects that have a matching **name** property
`type`	String Enum	Optional	Filter string to limit returned objects to those objects that have a matching **type** property. The value must be a valid **type** property value.
`cpc-name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching **name** property.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties (**name**, **object-uri**, **type**, **cpc-name**, **cpc-object-uri**). This is a list of comma-separated strings where each string is a property name defined in the Virtual Switch object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`virtual-switches`	Array of virtual-switch-info objects	Array of nested virtual-switch-info objects as described in the next table. The returned array may be empty.

Each nested virtual-switch-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The canonical URI path **object-uri** of the Virtual Switch object.
`name`	String	The **name** property of the Virtual Switch object
`type`	String Enum	The **type** property of the Virtual Switch object
`cpc-name`	String	The **name** property of the Virtual Switch's parent CPC object.
`cpc-object-uri`	String/ URI	The **object-uri** property of the Virtual Switch's parent CPC object. This property will be null when the current user has no object access permission to the CPC.
`object-id`	String	The unique identifier of te virtual switch instance.

Description

The `List Permitted Virtual Switches` operation lists virtual switches to which the API user has object-access permission. Some basic properties (**name**, **object-uri**, **type**, **cpc-name**, **cpc-object-uri**) are provided for each virtual switch that is included in the response

If the **additional-properties** query parameter is specified, the response body is enhanced with the additionally requested properties. The presence and value of each requested property is the same as it would be in the response body of a `Get Virtual Switch Properties` operation. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **additional-properties** query parameter is omitted, only the default properties are included in the response.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object identified in the request URI
- Object-access permission to the backing adapter of the Virtual Switch objects to be included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 412.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/list-permitted-virtual-switches HTTP/1.1
x-api-session: 16pgd58ffv61wq7bxrdy64jkstz1uhk3s9trjnjynz4xh38awa
```

Figure 169. List Permitted Virtual Switches: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Thu, 18 Jan 2024 10:35:14 GMT
Content-Type: application/json
Content-Length: 1007
{
   "virtual-switches":[
      {
         "cpc-name":"8X110741",
         "cpc-object-uri":"/api/cpcs/3ab48eb6-2b7e-3ded-89d0-64a68b209e75",
         "name":"14C.P0.OSD",
         "object-uri":"/api/virtual-switches/167d6a4c-b5ec-11ee-8f24-fa163ef1a60f",
         "type":"osd"
      },
      {
         "cpc-name":"8X110741",
         "cpc-object-uri":"/api/cpcs/3ab48eb6-2b7e-3ded-89d0-64a68b209e75",
         "name":"140.P0.OSD",
         "object-uri":"/api/virtual-switches/17f66b26-b5ec-11ee-8f24-fa163ef1a60f",
         "type":"osd"
      },
      {
         "cpc-name":"8X110741",
         "cpc-object-uri":"/api/cpcs/3ab48eb6-2b7e-3ded-89d0-64a68b209e75",
         "name":"104.P1.OSD",
         "object-uri":"/api/virtual-switches/13d97f42-b5ec-11ee-8f24-fa163ef1a60f",
         "type":"osd"
      },
      {
         "cpc-name":"8X110741",
         "cpc-object-uri":"/api/cpcs/3ab48eb6-2b7e-3ded-89d0-64a68b209e75",
         "name":"180.P0.OSD",
         "object-uri":"/api/virtual-switches/1eec1890-b5ec-11ee-8f24-fa163ef1a60f",
         "type":"osd"
      },
      {
         "cpc-name":"8X110741",
         "cpc-object-uri":"/api/cpcs/3ab48eb6-2b7e-3ded-89d0-64a68b209e75",
         "name":"104.P0.OSD",
         "object-uri":"/api/virtual-switches/13b641da-b5ec-11ee-8f24-fa163ef1a60f",
         "type":"osd"
      }
   ]
}
```

Figure 170. List Permitted Virtual Switches: Response

Get Virtual Switch Properties

The `Get Virtual Switch Properties` operation retrieves the properties of a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

```
GET /api/virtual-switches/{vswitch-id}
```

In this request, the URI variable *{vswitch-id}* is the object ID of the Virtual Switch object for which properties are to be returned.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Virtual Switch object as defined in the "Data model" on page 408. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current property values for the Virtual Switch object specified by *{vswitch-id}*.

On successful execution, all of the current properties as defined in the "Data model" on page 408 for the Virtual Switch object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Virtual Switch object and the API user must have object-access permission to the backing Adapter object of the virtual switch. If either of these conditions is not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified in the request URI

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 414.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the request URI *{vswitch-id}*) does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058
x-api-session: 5an6scz1o7mikkmyeew077vj9yggo5qe781wgegwpw3yilxq4j
```

Figure 171. Get Virtual Switch Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Mar 2015 07:00:51 GMT
content-type: 'application/json;charset=UTF-8
content-length: 306
{
 "backing-adapter-uri":"/api/adapters/f718c7a0-d490-11e4-a555-020000003058",
 "class":"virtual-switch",
 "description":"",
 "name":"PrimeIQDVSwitch1",
 "object-id":"f6b4c70e-d491-11e4-a555-020000003058",
 "object-uri":"/api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058",
 "parent":"/api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef",
 "port":0,
 "type":"hipersockets"
}
```

Figure 172. Get Virtual Switch Properties: Response

Get Connected VNICs of a Virtual Switch

The `Get Connected VNICs of a Virtual Switch` operation retrieves the list of network interfaces (VNICs) connected to a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

```
GET /api/virtual-switches/{vswitch-id}/operations/get-connected-vnics
```

In this request, the URI variable *{vswitch-id}* is the object ID of the Virtual Switch object whose VNIC list is to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the **connected-vnic-uris** property of the Virtual Switch object as defined in the "Data model" on page 408.

Description

This operation returns the list of VNICs connected to the Virtual Switch object specified by *{vswitch-id}*.

On successful execution, the current list of VNICs connected to the Virtual Switch is provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Virtual Switch object and the API user must have object-access permission to the backing Adapter object of the Virtual Switch object. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified by the request URI.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 416.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The object ID in the request URI (*{vswitch-id}*) does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058/operations/get-connected-
   vnics
x-api-session: 5an6scz1o7mikkmyeew077vj9yggo5qe781wgegwpw3yilxq4j
```

Figure 173. Get Connected VNICs of a Virtual Switch: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Mar 2015 07:00:51 GMT
content-type: 'application/json;charset=UTF-8
content-length: 370
{
 "connected-vnic-uris":[
 "/api/partitions/675fe728-dfc7-11e4-8582-020000003022/nics/0bc27850-dfc9-11e4-
    a45e-020000003022",
 "/api/partitions/675fe728-dfc8-11e4-8583-020000003022/nics/0bc27850-dfc9-11e4-
    a45e-020000003022",
 "/api/partitions/684fe825-dfc9-11e4-8681-020000003022/nics/0ba26450-dfc8-11e4-
    a36e-020000003022"
 ],
}
```

Figure 174. Get Connected VNICs of a Virtual Switch: Response

Update Virtual Switch Properties

The `Update Virtual Switch Properties` operation updates the properties of a single Virtual Switch object that is designated by its object ID.

HTTP method and URI

```
POST /api/virtual-switches/{vswitch-id}
```

In this request, the URI variable *{vswitch-id}* is the object ID of the Virtual Switch object for which properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for

this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

The `Update Virtual Switch Properties` operation updates writable properties of the Virtual Switch object specified by *{vswitch-id}*.

The URI path must designate an existing Virtual Switch object, and the API user must have object-access permission to the backing Adapter object of the Virtual Switch object. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Adapters** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in "Request body contents" on page 417. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the backing Adapter object of the Virtual Switch object specified in the request URI
- Action/task permission to the **Manage Adapters** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 177. Update Virtual Switch Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	API user does not have action/task permission to the **Manage Adapters** task.
404 (Not Found)	1	The object ID *{vswitch-id}* does not designate an existing Virtual Switch object, or the API user does not have object-access permission to the backing Adapter object of the Virtual Switch object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/virtual-switches/f6b4c70e-d491-11e4-a555-020000003058
x-api-session: rxkrzf6v287nn0klyk4zah94i91r49eskxkut8242kdqfa9so
content-type: application/json
content-length: 68
{
"description": "the neNetwork virtual switch",
"name": "neVswitch"
}
```

Figure 175. Update Virtual Switch Properties: Request

```
204 No Content
date: Tue, 31 Mar 2015 07:05:50 GMT
server: zSeries management console API web server / 2.0
cache-control: no-cache

<No response body>
```

Figure 176. Update Virtual Switch Properties: Response

Inventory service data

Information about the virtual switches can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Virtual Switch objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by inventory class, implicitly through a containing category, or by default) that objects of the class **"virtual-switch"** to be included. An entry for a particular virtual switch is included only if the API user has access permission to that object as described in the `Get Virtual Switch Properties` operation.

For each Virtual Switch object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Virtual Switch Properties" on page 414. That is, the data provided is the same as would be provided if a `Get Virtual Switch Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single virtual switch. This object would appear as one array entry in the response array:

```
{
  "backing-adapter-uri":"/api/adapters/df863694-08b1-11e6-91a3-42f2e90df693",
  "class":"virtual-switch",
  "description":"",
  "name":"5F1.P0.OSD",
  "object-id":"dff0b71c-d491-11e4-a555-020000003058",
  "object-uri":"/api/virtual-switches/dff0b71c-d491-11e4-a555-020000003058",
  "parent":"/api/cpcs/8e543aa6-1c26-3544-8197-4400110ef5ef",
  "port":0,
  "type":"osd"
}
```

Figure 177. Virtual Switch object: Sample inventory data - Response

Capacity Group element object

A Capacity Group is an element object of a CPC that is in DPM mode. It consists of a set of partitions and specifies the absolute processor cap per processor type for that set of partitions. The defined absolute processor cap for a specific processor type dictates the total amount of the specified processing capacity that the active partitions in the group can consume at any time. The absolute processor cap is specified as a value between 0.01 and 255.0, where a value of 1.00 represents the processing capacity provided by one processor.

The scope of a Capacity Group is within a single CPC and it is required that the CPC is in DPM mode for a Capacity Group to be created.

The absolute processor cap can be modified after the group is created, at which time, the partitions that are already active in the group will be affected.

Data model

The following attributes are identified for the Capacity Group element object:

Table 178. Capacity Group element object properties

Name	Qualifier	Type	Description of specialization
element-uri	—	String/ URI	The canonical URI path of the Capacity Group element object is of the form `/api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}` where `{capacity-group-id}` is the unique identifier for a capacity group within a CPC.
element-id	—	String (36)	The unique identifier for the capacity group. The **element-id** is in the form of a UUID.
parent	—	String/ URI	The canonical URI path of the parent CPC object.
class	—	String (14)	The class of a Capacity Group object is **"capacity-group"**
name	(w)(pc)	String (1-64)	The name of the capacity group. The name must be unique among all capacity groups in the CPC. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100.
short-name	(w)(pc)	String (1-8)	The name of the capacity group that will be presented to the guest OS, when capping is enabled on this capacity group. Only alpha-numeric uppercase characters are allowed. The **short-name** must be unique among all capacity groups in the CPC. The words PHYSICAL, REC, SYSTEM, and PRIM*nnnn* (where *nnnn* is a 4-digit number) are reserved and cannot be used.
description	(w)(pc)	String (0-1024)	The description of the capacity group. Default: An empty string.

Table 178. Capacity Group element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
capping-enabled	(w)(pc)	Boolean	This indicates if capping is enabled for this capacity group or not. If set to true, the partitions in this capacity group are capped according to the values set. If set to false, any specified cap values are not effective and the partitions in this capacity group are not capped. Default: True.
absolute-general-purpose-proc-cap	(w)(pc)	Float (0-255)	The limit on the absolute capacity of general purpose processors that the partitions in this group are allowed to consume at any point in time. It is expressed in the units of processors. The range of valid values is between 0.0 and 255.0 and in the increments of 0.01. Default: 0.0. The value of 0.0 indicates that the general purpose processors are not capped for the partitions in this capacity group. Exactly one of the processor types must have a value other than 0.0.
absolute-ifl-proc-cap	(w)(pc)	Float (0-255)	The limit on the absolute capacity of IFL processors that the partitions in this group are allowed to consume at any point in time. It is expressed in the units of processors. The range of valid values is between 0.0 and 255.0 and in the increments of 0.01. Default: 0.0. The value of 0.0 indicates that the IFL processors are not capped for the partitions in this capacity group. Exactly one of the processor types must have a value other than 0.0.
partition-uris	(c)(pc)	Array of String/ URIs	Array of URIs of the partitions that belong to this capacity group. Each partition in this capacity group must have the type of processor (general purpose or IFL) that is being capped by this capacity group. This property can be modified using the `Add Partition to Capacity Group` and `Remove Partition from Capacity Group` operations.

List Capacity Groups of a CPC

The `List Capacity Groups of a CPC` operation lists the defined capacity groups for the specified CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-groups
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
capacity-groups	Array of capacity-group-info objects	Array of capacity-group-info objects, described in the next table. Returned array may be empty.

Each nested capacity-group-info object contains the following fields:

Field name	Type	Description
element-uri	String/ URI	Canonical URI path of the Capacity Group object. More details about this capacity group can be fetched using this **element-uri**.
name	String (1-64)	The name of the capacity group.

Description

This operation lists the capacity groups that are defined for the specified CPC. The **element-uri** and **name** are returned for each capacity group.

If the **name** query parameter is specified for the request, the returned list is limited to the Capacity Group elements that have a name matching the specified filter pattern. If no match is found, then the response will be an empty array. If the parameter is not specified, all the Capacity Group elements are returned.

The response could be an empty array, if the CPC does not have a Capacity Group associated with it or if the CPC is not in DPM mode.

The URI path must designate an existing CPC, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the specified CPC.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 422.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not exist or the user does not have object-access permission to it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups HTTP/1.1
x-api-session: 4xjbyzz8iip3fv0j77gyuk3e6r7lp0pl9yb4zfpf1n46u3z7ec
```

Figure 178. List Capacity Groups of a CPC: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 22 Feb 2016 10:00:54 GMT
content-type: application/json;charset=UTF-8
content-length: 363
{
   "capacity-groups":[
      {
         "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/e0118d40-
            d088-11e5-a631-42f2e9ef1641",
         "name":"Test18311"
      },
      {
         "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/d95f6418-
            d6eb-11e5-92b9-42f2e9ef1641",
         "name":"myNewGroup1"
      }
   ]
}
```

Figure 179. List Capacity Groups of a CPC: Response

Create Capacity Group

The `Create Capacity Group` operation creates a capacity group for the specified CPC.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/capacity-groups
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
name	String (1-64)	Required	The value to be set as the Capacity Group's **name** property.

Field name	Type	Rqd/Opt	Description
`short-name`	String (1-8)	Optional	The value to be set as the Capacity Group's **short-name** property. If a **short-name** is not specified, it will be auto-generated and assigned to this capacity group. The name, if specified, must be unique within the CPC.
`description`	String (0-1024)	Optional	The value to be set as the Capacity Group's **description** property.
`capping-enabled`	Boolean	Optional	The value to be set as the capacity group's **capping-enabled** property.
`absolute-general-purpose-proc-cap`	Float	Optional	The value to be set as the Capacity Group's **absolute-general-purpose-proc-cap** property.
`absolute-ifl-proc-cap`	Float	Optional	The value to be set as the Capacity Group's **absolute-ifl-proc-cap** property.

Response body contents

On successful completion, HTTP status code 201 (Created) is returned and a JSON object with the following field is also provided. The **element-uri** of the capacity group created is also available in the Location header of the response.

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the created capacity group.

Description

This operation creates a capacity group with the specified attributes. It is required that the CPC designated by *{cpc-id}* is in DPM mode. On successful execution, the **element-uri** of the created capacity group is returned in the response body and in the **Location** response header. An Inventory Change notification is emitted asynchronously to this operation.

Exactly one processor type must have an absolute processor cap other than 0.0 and within the valid range of 0.01 and 255.0. Otherwise, a 400 (Bad Request) status code is returned.

If the *{cpc-id}* does not designate a CPC that is in DPM mode, an HTTP status code 409 (Conflict) is returned.

The URI path must designate an existing CPC, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

If the request body contents fail to validate, a 400 (Bad Request) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 424.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 179. Create Capacity Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permission to the **Manage Processor Sharing** task.
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not exist or the user does not have object-access permission to the CPC.
409 (Conflict)	5	The CPC is not in DPM mode.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups HTTP/1.1
x-api-session: jht693kvysnx78mws1i0sacatmvougd2y11jz0e0jmgy3fv94
content-type: application/json
content-length: 160
{
   "absolute-general-purpose-proc-cap":5.0,
   "capping-enabled":true,
   "description":"Test Group for CP Procs",
   "name":"Cap group 123",
   "short-name":"CPGRP111"
}
```

Figure 180. Create Capacity Group: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/a4d8826c-
   d9fa-11e5-8b15-42f2e9ef1641
cache-control: no-cache
date: Tue, 23 Feb 2016 07:00:57 GMT
content-type: application/json;charset=UTF-8
content-length: 117
{
   "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-
      d9fb-11e5-8b3e-42f2e9ef1641"
}
```

Figure 181. Create Capacity Group: Response

Delete Capacity Group

The `Delete Capacity Group` operation deletes the specified capacity group.

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Description

This operation deletes the capacity group with the specified *{capacity-group-id}*. It is required that the specified capacity group does not contain any partition, at the time of deletion. An Inventory Change notification is emitted asynchronously to this operation.

The URI path must designate an existing Capacity Group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

A 409 (Conflict) status code is returned if the capacity group contains any partitions. The `Remove Partition from Capacity Group` operation should be used to delete all the partitions from this capacity group before attempting this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 180. Delete Capacity Group: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Processor Sharing** task
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not exist or the user does not have object-access permission to the CPC.
	150	The capacity group identified by *{capacity-group-id}* does not exist.
409 (Conflict)	110	The capacity group contains one or more partitions, and thus could not be deleted.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/a4d8826c-d9fa-
   11e5-8b15-42f2e9ef1641 HTTP/1.1
x-api-session: 2saqzhzg358oqoke89osqrgt09k27rqwy4d0mtdp6j2z05ok6q
```

Figure 182. Delete Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:02:43 GMT
<No response body>
```

Figure 183. Delete Capacity Group: Response

Get Capacity Group Properties

The `Get Capacity Group Properties` operation retrieves the properties of a single capacity group.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Response body contents

On successful completion, an HTTP status code 200 (OK) is returned and a JSON object containing the current values of the properties for the Capacity Group element object as defined in the "Data model" on page 420 is provided as a response body. Field names and data types in the JSON object are the same as the property names and data types that are defined in the data model.

Description

This operation retrieves the current values of the properties of the capacity group.

The URI path must designate an existing Capacity Group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the specified CPC.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and a response body with the current values for the properties of the capacity group is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 181. Get Capacity Group Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The object ID *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to it.
	150	The capacity group identified by *{capacity-group-id}* does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-
   8b3e-42f2e9ef1641 HTTP/1.1
x-api-session: 487yumkfzjybynsm00a5qn7v916s8eq49krcnpfqut4n0kfv4
```

Figure 184. Get Capacity Group Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:14:23 GMT
content-type: application/json;charset=UTF-8
content-length: 504
{
   "absolute-general-purpose-proc-cap":5.0,
   "absolute-ifl-proc-cap":0.0,
   "capping-enabled":true,
   "class":"capacity-group",
   "description":"Test Group for CP Procs",
   "element-id":"1de263e4-d9fb-11e5-8b3e-42f2e9ef1641",
   "element-uri":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-
      d9fb-11e5-8b3e-42f2e9ef1641",
   "name":"Cap group 123",
   "parent":"/api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0",
   "partition-uris":[
      "/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
   ],
   "short-name":"CPGRP111"
}
```

Figure 185. Get Capacity Group Properties: Response

Add Partition to Capacity Group

The `Add Partition to Capacity Group` operation adds a partition to the specified capacity group.

HTTP method and URI

`POST /api/cpcs/`*`{cpc-id}`*`/capacity-groups/`*`{capacity-group-id}`*`/operations/add-partition`

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object with the following field:

Name	Type	Rqd/Opt	Description
`partition-uri`	String/ URI	Required	The canonical URI of the Partition object to be added to the capacity group.

Description

This operation adds a partition to an existing capacity group identified by the *{capacity-group-id}*.

A partition can be added to the capacity group, only if the processor type that the partition uses has a nonzero cap value in the specified capacity group. Otherwise, a status code 409 (Conflict) is returned.

A capacity group can only contain partitions with shared processors. If the partition specified in the request is configured for dedicated processors, a status code 409 (Conflict) will be returned.

A partition cannot become a member of more than one capacity group. If the partition specified in the request is currently a member of another capacity group or the capacity group identified by *{capacity-group-id}*, a status code 409 (Conflict) is returned.

If the partition does not belong to the same CPC as the capacity group, a status code 400 (Bad Request) is returned.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

On successful execution, the partition is added to the capacity group and the number of processors that could be used by this partition becomes governed by the absolute proc cap values that are defined for this capacity group.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 182. Add Partition to Capacity Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	160	The partition does not belong to the same CPC as the capacity group.
403 (Forbidden)	1	API user does not have action/task permission to the **Manage Processor Sharing** task.
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not have object-access permission to the CPC.
	2	The partition designated by the **partition-uri** in the request body does not exist.
	150	The capacity group identified by *{capacity-group-id}* does not exist.

Table 182. Add Partition to Capacity Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	120	The partition is currently a member of another capacity group and thus, cannot be added to this capacity group.
	130	The partition is already a member of the capacity group identified by *{capacity-group-id}*.
	170	The partition is configured with dedicated processors.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-
   8b3e-42f2e9ef1641/operations/add-partition HTTP/1.1
x-api-session: 3ty54o1c7961o2jt0jyjm5nm8xqxofys7s302syjoguaycbeo5
content-type: application/json
content-length: 73
{
   "partition-uri":"/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
}
```

Figure 186. Add Partition to Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:13:42 GMT
<No response body>
```

Figure 187. Add Partition to Capacity Group: Response

Remove Partition from Capacity Group

The `Remove Partition from Capacity Group` operation removes a partition from an existing capacity group.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}/operations/remove-partition
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.

Name	Description
{capacity-group-id}	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`partition-uri`	String/ URI	Required	The canonical URI of the Partition object to be removed from the capacity group.

Description

This operation removes a partition from a capacity group identified by the *{capacity-group-id}*. If the partition is not currently a member of the capacity group, a status code 409 (Conflict) is returned.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

On successful execution, the partition is removed from the capacity group and the amount of processing capacity that could be used by this partition is no longer governed by the absolute proc cap values defined for this capacity group.

If this operation changes the values of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned, and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 183. Remove Partition from Capacity Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permission to the **Manage Processor Sharing** task.

Table 183. Remove Partition from Capacity Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not exist or the user does not have object-access permission to the CPC.
	2	The partition designated by the **partition-uri** in the request body does not exist.
	150	The capacity group identified by *{capacity-group-id}* does not exist.
409 (Conflict)	140	The partition is not currently a member of the capacity group.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-
   8b3e-42f2e9ef1641/operations/remove-partition HTTP/1.1
x-api-session: 3581gp14ocfmtodeh93ckw1ita2c3k07mrv6xb7ymqgcncjv8k
content-type: application/json
content-length: 73
{
   "partition-uri":"/api/partitions/880c3272-cbeb-11e5-90fe-42f2e9ef1641"
}
```

Figure 188. Remove Partition from Capacity Group: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:16:00 GMT
<No response body>
```

Figure 189. Remove Partition from Capacity Group: Response

Update Capacity Group Properties

The `Update Capacity Group Properties` operation modifies the writable properties of a capacity group.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/capacity-groups/{capacity-group-id}
```

URI variables:

Name	Description
{cpc-id}	Object ID of the CPC.
{capacity-group-id}	Element ID of the capacity group.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the Capacity Group element specified by the *{capacity-group-id}*.

The URI path must designate an existing capacity group, and the API user must have object-access permission to the CPC object specified by the *{cpc-id}*. If these conditions are not met, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Processor Sharing** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the "Data model" on page 420. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The capacity group must be removed of all partitions first, if the cap values are updated such that the capacity group starts capping partitions on a different processor type. Otherwise, a status code 400 (Bad Request) is returned.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the specified CPC.
- Action/task permission to the **Manage Processor Sharing** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 184. Update Capacity Group Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Processor Sharing** task.

Table 184. Update Capacity Group Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The CPC identified by *{cpc-id}* does not exist or the user does not have object-access permission to the CPC.
	150	The capacity group identified by *{capacity-group-id}* does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/3baea1ec-76e8-3e42-a111-815a7aee19e0/capacity-groups/1de263e4-d9fb-11e5-
   8b3e-42f2e9ef1641 HTTP/1.1
x-api-session: 8rws222hts1w985vhnn5vcne4x5z6b3q143z0gx7kqzmnuqlu
content-type: application/json
content-length: 42
{
   "absolute-general-purpose-proc-cap":2.0
}
```

Figure 190. Update Capacity Group Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 23 Feb 2016 07:18:42 GMT
<No response body>
```

Figure 191. Update Capacity Group Properties: Response

Inventory service data

Information about capacity groups can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Capacity Group objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular capacity group is included only if the API user has access permission to that object as described in the `Get Capacity Group Properties` operation.

For each Capacity Group object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Capacity Group Properties" on page 427. That is, the data provided is the same as would be provided if a `Get Capacity Group Properties` operation were requested targeting this object.

Storage Site object

A Storage Site object represents a single storage site in the FICON storage configuration associated with a DPM-enabled CPC. A storage site describes a location that houses a set of storage switches and storage subsystems. A primary site with a default name of "Primary Site", local to the CPC, exists by default and cannot be deleted. An alternate site, typically at a remote location, can be created. The Storage Site object APIs provide access to the set of storage sites within the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to create and delete alternate storage sites, query storage sites, and update selected properties of storage sites.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 186 on page 436. The Storage Site object does not support the operational status related properties.

Table 185. Storage Site object properties

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Site object is of the form `/api/storage-sites/{storage-site-id}` where `{storage-site-id}` is the value of the **object-id** property of the Storage Site object.
object-id	—	String (36)	The unique identifier for the storage site instance.
class	—	String (12)	The class of the Storage Site object is **"storage-site"**.
parent	—	String/ URI	The parent of a storage site is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
name	(w)(pc)	String (1-64)	The display name specified for the storage site. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage sites within the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** property.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage site. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 186. Storage Site object: class specific properties

Name	Qualifier	Type	Description
cpc-uris	—	Array of String/ URI	The list of CPCs that reside in this storage site. Each element in this array is the canonical URI path of a CPC.

Table 186. Storage Site object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
storage-subsystem-uris	(c)(pc)	Array of String/ URI	The list of storage subsystems that reside in this storage site. Each element in this array is the canonical URI path of a Storage Subsystem object. The value of this property will change, and property change notifications will be emitted, when storage subsystem objects are defined or undefined.
storage-switch-uris	(c)(pc)	Array of String/ URI	The list of storage switches that reside in this storage site. Each element in this array is the canonical URI path of a Storage Switch object. The value of this property will change, and property change notifications will be emitted, when storage switch objects are defined or undefined.
type	—	String Enum	The type of this storage site. Values: • **"primary"** - The storage site is the primary, typically local, location. • **"alternate"** - The storage site is an alternate, typically remote, location.

List Storage Sites

The `List Storage Sites` operation lists the storage sites known to the target Console.

HTTP method and URI

```
GET /api/storage-sites
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching CPC URI in its **cpc-uris** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-sites`	Array of storage-site-info objects	Array of storage-site-info objects, described in the next table. The returned array may be empty.

Each nested storage-site-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Storage Site object.
`name`	String	The **name** property of the Storage Site object.
`cpc-uris`	Array of String/ URI	The **cpc-uris** property of the Storage Site object.
`type`	String Enum	The **type** property of the Storage Site object.

Description

This operation lists the storage sites that are known to the target Console. The object URI, name, type and CPC URI list are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage sites that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid value for the storage site **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage sites that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not done.

If the **cpc_uri** query parameter is specified, the returned list is limited to those storage sites that have a matching CPC URI in its **cpc-uris** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage site is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the API user does not have permission to a storage site, that object is simply omitted from the list but no error status code results.

If no storage sites are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 437.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 187. List Storage Sites: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-sites HTTP/1.1
x-api-session: 1exubjc30uxq0rpf2gyycx230m9voc6rpuqperj8gdya59d422
```

Figure 192. List Storage Sites: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:08:43 GMT
content-type: application/json;charset=UTF-8
content-length: 363
{
   "storage-sites":[
      {
         "cpc-uris":[
            "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
         ],
         "name":"New York",
         "object-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492",
         "type":"primary"
      },
      {
         "cpc-uris":[
            "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
         ],
         "name":"New Jersey",
         "object-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492",
         "type":"alternate"
      }
   ]
}
```

Figure 193. List Storage Sites: Response

Create Storage Site

The `Create Storage Site` operation creates a new alternate Storage Site object.

HTTP method and URI

```
POST /api/storage-sites
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uris`	Array of String/ URI	Required	The value to be set as the storage site's **cpc-uris** property. Currently, this list must contain a single CPC URI, representing the CPC associated with the FICON configuration to which this storage site belongs. This limitation may change in the future.
`name`	String (1-64)	Required	The value to be set as the storage site's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the storage site's **description** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the newly created Storage Site object.

Description

This operation creates an alternate storage site with the values specified and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the **cpc-uris** field in the request body contains a URI that does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** field in the request body already contains a storage site with the specified name, a 400 (Bad Request) status code is returned. If the CPC identified by the URI in the **cpc-uris** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled, or if its FICON configuration already contains the maximum number of alternate storage sites, or the **cpc-uris** field does not contain exactly one CPC URI, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage site is created and its properties are set to their corresponding request body content's field's values. The new storage site's **type** property is set to **"alternate"**. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to each CPC whose **object-uri** is in the **cpc-uris** list.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 440.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 188. Create Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage site with the **name** specified in the request body already exists within the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** field specified in the request body.

Table 188. Create Storage Site: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A CPC identified by a URI in the **cpc-uris** field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	5	The CPC identified by the URI in the **cpc-uris** field is not enabled for DPM.
	13	A CPC identified by a URI in the **cpc-uris** field does not support the **dpm-storage-management** feature.
	329	The operation cannot be performed because the CPC identified by the URI in the **cpc-uris** field is an unmanaged CPC, which is not supported by this operation.
	440	The maximum number of alternate storage sites, defined by the CPC **maximum-alternate-storage-sites** property, already exist within the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** field in the request body.
	451	The **cpc-uris** field does not contain exactly one CPC URI.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-sites HTTP/1.1
x-api-session: 4htyri6wi6g3tr91kv01hqxdn9aiucykq3e0ne21uvepky34yp
content-type: application/json
content-length: 119
{
   "cpc-uris":[
      "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
   ],
   "description":"Alternate site",
   "name":"New Jersey"
}
```

Figure 194. Create Storage Site: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492
cache-control: no-cache
date: Mon, 30 Jul 2018 20:06:13 GMT
content-type: application/json;charset=UTF-8
content-length: 72
{
   "object-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 195. Create Storage Site: Response

Usage notes

Each CPC maintains its own view of the physical storage sites. When a list operation is targeted at a console, there will likely be multiple storage site instances returned that represent the same physical site – one for each CPC to which that physical site is configured. There is no intrinsic storage site property that can be used to correlate storage site instances that represent the same physical site. It is therefore recommended that API clients adopt a naming convention that ensures storage sites that represent the same physical site have the same value of their respective **name** properties.

FICON configurations are currently limited to a single alternate storage site. This may change in the future, it which case a console could be managing different versions of CPC that have different alternate storage site limits. It is recommended that API clients use the **maximum-alternate-storage-sites** property of the CPC object to programmatically determine how many alternate sites can be created.

Delete Storage Site

The `Delete Storage Site` operation deletes an alternate storage site.

HTTP method and URI

```
DELETE /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the alternate storage site to delete.

Description

This operation deletes an alternate storage site. The storage site must be empty; it cannot contain any storage switches or storage subsystems. An Inventory Change notification for the deleted storage site is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-site-id}* does not identify a storage site object on the HMC. If the value of the target storage site's **type** property value is **"primary"**, a 400 (Bad Request) status code is returned. If the target storage site's **storage-fabric-uris** or **storage-switch-uris** array properties are not empty, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage site is deleted from the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 189. Delete Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	440	The storage site object with the **object-id** *{storage-site-id}* has a **type** property value of **"primary"**.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage site with the **object-id** *{storage-site-id}* does not exist on the HMC.
409 (Conflict)	2	The storage site object with the **object-id** *{storage-site-id}* was busy and the request timed out.
	452	The storage site with the object-id *{storage-site-id}* has a **storage-fabric-uris** or **storage-switch-uris** property that is not empty.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: ynux0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
```

Figure 196. Delete Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 197. Delete Storage Site: Response

Get Storage Site Properties

The `Get Storage Site Properties` operation retrieves the properties of a single Storage Site object.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the storage site object.

Response body contents

On successful completion, an HTTP status code 200 (OK) is returned and a JSON object containing the current values of the properties for the Storage Site object as defined in the "Data model" on page 436 is provided as a response body. Field names and data types in the JSON object are the same as the property names and data types that are defined in the data model.

Description

Returns the current values of the properties for the storage site object as defined in the "Data model" on page 436 .

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-site-id}* does not identify a storage site object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 444.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 190. Get Storage Site Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage site with **object-id** *{storage-site-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492 HTTP/1.1
x-api-session: 46y82ucgax84e1jtyy5w5ugv1l9lvpewu2xp9d865ipz4skwqo
```

Figure 198. Get Storage Site Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:12:31 GMT
content-type: application/json;charset=UTF-8
content-length: 595
{
   "class":"storage-site",
   "cpc-uris":[
      "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
   ],
   "description":"Primary site",
   "name":"New York",
   "object-id":"13ff101c-941f-11e8-a0c0-fa163e27d492",
   "object-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492",
   "parent":"/api/console",
   "storage-subsystem-uris":[
      "/api/storage-subsystems/37af8766-943e-11e8-8ffe-fa163e27d492",
      "/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
   ],
   "storage-switch-uris":[
      "/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
      "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
   ],
   "type":"primary"
}
```

Figure 199. Get Storage Site Properties: Response

Update Storage Site Properties

The `Update Storage Site Properties` operation updates one or more of the writable properties of a storage site.

HTTP method and URI

```
POST /api/storage-sites/{storage-site-id}
```

In this request, the URI variable *{storage-site-id}* is the object ID of the Storage Site object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 436. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage site's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-site-id}* does not identify a Storage Site object on

the HMC. If the FICON configuration associated with the CPC identified by the URI in the storage site's **cpc-uris** property already contains a storage site with the specified name, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage site's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 191. Update Storage Site Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage site with the **name** specified in the request body already exists within the FICON configuration associated with the CPC identified by the URI in the **cpc-uris** property of the storage site with the **object-id** *{storage-site-id}*.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage site with the **object-id** *{storage-site-id}* does not exist on the HMC.
409 (Conflict)	2	The storage site object with the **object-id** *{storage-site-id}* was busy and the request timed out.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492 HTTP/1.1
x-api-session: 1uhgm42bm5vbmtbwrvvpqqic5u3ffc4aqz40cjwtsjlcpm042
content-type: application/json
content-length: 51
{
   "description":"Primary site",
   "name":"New York"
}
```

Figure 200. Update Storage Site Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:08:16 GMT

<No response body>
```

Figure 201. Update Storage Site Properties: Response

Inventory service data

Information about the Storage Sites managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Storage Site objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-site" are to be included. Information for a particular storage site is included only if the API user has access permission to that object as described in the `Get Storage Site Properties` operation.

For each storage site to be included, the inventory response includes an array entry for the Storage Site object. This entry is a JSON object with the same contents as is specified in the Response body contents section of "Get Storage Site Properties" on page 444. That is, the data provided is the same as would be provided if a `Get Storage Site Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a storage site. This object would appear as one array entry in the response array:

```
{
   "class":"storage-site",
   "cpc-uris":[
      "/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e"
   ],
   "description":"Primary site",
   "name":"New Jersey",
   "object-id":"0336a208-9434-11e8-9c43-fa163e27d492",
   "object-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492",
   "parent":"/api/console",
   "storage-subsystem-uris":[
      "/api/storage-subsystems/9b15669a-943e-11e8-8ffe-fa163e27d492",
      "/api/storage-subsystems/93791760-943e-11e8-9c43-fa163e27d492"
   ],
   "storage-switch-uris":[
      "/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
      "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492"
   ],
   "type":"alternate"
}
```


Figure 202. Storage Site object: Sample inventory data - Response

Storage Fabric object

A Storage Fabric object represents a single storage fabric in the FICON configuration associated with a DPM-enabled CPC. A storage fabric is a collection of interconnected storage switches. If the storage configuration contains multiple storage sites, a storage fabric can, and typically does, span those sites. The Storage Fabric object APIs provide access to the set of storage fabrics within the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to create and delete storage fabrics, list storage fabrics, query storage fabric properties, and update selected properties of storage fabrics.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 193 on page 449. The storage fabric object does not support the operational status related properties.

Table 192. Storage Fabric object properties

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Fabric object is of the form `/api/storage-fabrics/{storage-fabric-id}` where *{storage-fabric-id}* is the value of the **object-id** property of the Storage Fabric object.
object-id	—	String (36)	The unique identifier for the storage fabric instance.
class	—	String (14)	The class of the Storage Fabric object is **"storage-fabric"**.
parent	—	String/ URI	The parent of a storage fabric is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.

Table 192. Storage Fabric object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the storage fabric. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage fabrics within the FICON configuration associated with the CPC identified by the URI in the **cpc-uri** property.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage fabric. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 193. Storage Fabric object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with the FICON configuration in which this storage fabric object resides.
storage-switch-uris	(c)(pc)	Array of String/ URI	The list of storage switches that comprise this storage fabric. Each element in this array is the canonical URI path of a Storage Switch object. The value of this property will change, and property change notifications will be emitted, when storage switch objects are defined or undefined.
high-integrity	(w)(pc)	Boolean	Indicates whether this fabric had been configured as a high integrity fabric. If this fabric contains switches that exist in different storage sites, this value will be **true** and cannot be written. Default value: **false**

List Storage Fabrics

The `List Storage Fabrics` operation lists the storage fabrics known to the target Console.

HTTP method and URI

```
GET /api/storage-fabrics
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
storage-fabrics	Array of storage-fabric-info objects	Array of storage-fabric-info objects, described in the next table. The returned array may be empty.

Each nested storage-fabric-info object contains the following fields:

Field name	Type	Description
object-uri	String/URI	Canonical URI path (**object-uri**) of the Storage Fabric object.
name	String	The **name** property of the Storage Fabric object.
cpc-uri	String/URI	The **cpc-uri** property of the Storage Fabric object.

Description

This operation lists the storage fabrics that are known to the target Console. The object URI, name, and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage fabrics that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage fabrics that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage fabric is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the API user does not have permission to a storage fabric, that object is simply omitted from the list but no error status code results.

If no storage fabrics are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 450.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 194. List Storage Fabrics: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-fabrics HTTP/1.1
x-api-session: 3tmygsxg76gvwqr4bgf2kpm2aj7z8nrbjb9ffiw6hhxfmjpm9b
```

Figure 203. List Storage Fabrics: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:22:26 GMT
content-type: application/json;charset=UTF-8
content-length: 325
{
   "storage-fabrics":[
      {
         "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
         "name":"Fabric B",
         "object-uri":"/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492"
      },
      {
         "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
         "name":"Fabric A",
         "object-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
      }
   ]
}
```

Figure 204. List Storage Fabrics: Response

Create Storage Fabric

The `Create Storage Fabric` operation creates a new Storage Fabric object.

HTTP method and URI

```
POST /api/storage-fabrics
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The value to be set as the storage fabric's **cpc-uri** property.

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the storage fabric's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the storage fabric's **description** property.
`high-integrity`	Boolean	Optional	The value to be set as the storage fabric's **high-integrity** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The Object URI of the newly created Storage Fabric object.

Description

This operation creates an alternate storage fabric with the values specified and then returns its **object-uri** in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the **cpc-uri** field in the request body contains a URI that does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. If the FICON configuration associated with the CPC identified by the **cpc-uri** field in the request body already contains a storage fabric with the specified name, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled, or if direct connections between storage subsystems and adapter ports exist, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage fabric is created and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 452.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 195. Create Storage Fabric: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage fabric with the **name** specified in the request body already exists within the FICON configuration associated with the CPC identified by the **cpc-uri** specified in the request body.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A CPC identified by the **cpc-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	5	The CPC identified by the **cpc-uri** field is not enabled for DPM.
	13	The CPC identified by the **cpc-uri** field does not support the **dpm-storage-management** feature.
	329	The operation cannot be performed because the CPC identified by the **cpc-uri** field is an unmanaged CPC, which is not supported by this operation.
	455	One or more direct physical connections between a storage subsystem and an adapter port exist.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-fabrics HTTP/1.1
x-api-session: 1r6vll1yh6tywdhmm5b7jvgf7zyf1g5r5ofs7tmmwcp6a58sx
content-type: application/json
content-length: 81
{
   "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
   "name":"Fabric A"
}
```

Figure 205. Create Storage Fabric: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492
cache-control: no-cache
date: Mon, 30 Jul 2018 20:20:41 GMT
content-type: application/json;charset=UTF-8
content-length: 74
{
   "object-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
}
```

Figure 206. Create Storage Fabric: Response

Usage notes

Each CPC maintains its own view of the physical storage fabric. When a list operation is targeted at a console, there will likely be multiple storage fabric instances returned that represent the same physical fabric – one for each CPC to which that physical fabric is configured. There is no intrinsic storage fabric property that can be used to correlate storage fabric instances that represent the same physical fabric. It is therefore recommended that API clients adopt a naming convention that ensures storage fabrics that represent the same physical fabric have the same value of their respective **name** properties.

Delete Storage Fabric

The `Delete Storage Fabric` operation deletes a storage fabric.

HTTP method and URI

```
DELETE /api/storage-fabrics/{storage-fabric-id}
```

In this request, the URI variable *{storage-fabric-id}* is the object ID of the alternate storage fabric to delete.

Description

This operation deletes a storage fabric. The storage fabric must be empty; it cannot contain any storage switches. An Inventory Change notification is emitted for the deleted storage fabric asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-fabric-id}* does not identify a storage fabric object on the HMC. If the target storage fabric's **storage-switch-uris** array property is not empty, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage fabric is deleted from the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 196. Delete Storage Fabric: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage fabric with the **object-id** *{storage-fabric-id}* does not exist on the HMC.
409 (Conflict)	2	The storage fabric object with the **object-id** *{storage-fabric-id}* was busy and the request timed out.
	452	The storage fabric with the object-id *{storage-site-id}* has a **storage-switch-uris** property that is not empty.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: ynux0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
```

Figure 207. Delete Storage Fabric: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 208. Delete Storage Fabric: Response

Get Storage Fabric Properties

The `Get Storage Fabric Properties` operation retrieves the properties of a single Storage Fabric object.

HTTP method and URI

```
GET /api/storage-fabrics/{storage-fabric-id}
```

In this request, the URI variable *{storage-fabric-id}* is the object ID of the storage fabric object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Fabric object as defined in the "Data model" on page 448. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Fabric object as defined in the "Data model" on page 448.

If the API user does not have action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-fabric-id}* does not identify a Storage Fabric object on the HMC, a 404 (Not Found) status code is returned. [Updated by feature **dpm-smcd-partition-link-management**]

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks. [Updated by feature **dpm-smcd-partition-link-management**]

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 456.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 197. Get Storage Fabric Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks. [Updated by feature **dpm-smcd-partition-link-management**]
404 (Not Found)	1	A storage fabric with **object-id** *{storage-fabric-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: knksjscdlzcrrh5le46x387k3uqk8qkp1tahyacguhq0p7m0d
```

Figure 209. Get Storage Fabric Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:03:05 GMT
content-type: application/json;charset=UTF-8
content-length: 452
{
   "class":"storage-fabric",
   "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
   "description":"Storage fabric A",
   "high-integrity":true,
   "name":"Fabric A",
   "object-id":"08ad557c-9436-11e8-9c43-fa163e27d492",
   "object-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
   "parent":"/api/console",
   "storage-switch-uris":[
      "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
      "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
   ]
}
```

Figure 210. Get Storage Fabric Properties: Response

Update Storage Fabric Properties

The `Update Storage Fabric Properties` operation updates one or more of the writable properties of a storage fabric.

HTTP method and URI

```
POST /api/storage-fabrics/{storage-fabric-id}
```

In this request, the URI variable *{storage-fabric-id}* is the object ID of the Storage Fabric object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 448. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage fabric's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-fabric-id}* does not identify a Storage Fabric object on the HMC. If the FICON configuration associated with the CPC identified by the URI in the storage fabric's **cpc-uri** property already contains a storage fabric with the specified name, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage fabric's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 198. Update Storage Fabric Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage fabric with the **name** specified in the request body already exists within the FICON configuration associated with the CPC identified by the **cpc-uri** property of the storage fabric with the **object-id** *{storage-fabric-id}*.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage fabric with the **object-id** *{storage-fabric-id}* does not exist on the HMC.
409 (Conflict)	2	The storage fabric object with the **object-id** *{storage-fabric-id}* was busy and the request timed out.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 3cqqq1phyn2nepyewzvway545dpi8z5tycmowv3gqf5nvbw107
content-type: application/json
content-length: 35
{
   "description":"Storage fabric A"
}
```

Figure 211. Update Storage Fabric Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:25:17 GMT

<No response body>
```

Figure 212. Update Storage Fabric Properties: Response

Inventory service data

Information about the Storage Fabrics managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage fabric objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-fabric" are to be included. Information for a particular storage fabric is included only if the API user has access permission to that object as described in the `Get Storage Fabric Properties` operation.

For each storage fabric to be included, the inventory response includes an array entry for the Storage Fabric object. This entry is a JSON object with the same contents as is specified in the Response body contents section of “Get Storage Fabric Properties” on page 455. That is, the data provided is the same as would be provided if a `Get Storage Fabric Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a storage fabric. This object would appear as one array entry in the response array:

```
{
   "class":"storage-fabric",
   "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
   "description":"Storage fabric A",
   "high-integrity":true,
   "name":"Fabric A",
   "object-id":"08ad557c-9436-11e8-9c43-fa163e27d492",
   "object-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
   "parent":"/api/console",
   "storage-switch-uris":[
      "/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
      "/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
   ]
}
```

Figure 213. Storage Fabric object: Sample inventory data - Response

Storage Switch object

A Storage Switch object represents a single storage switch in the FICON configuration associated with a DPM-enabled CPC. The Storage Switch object APIs provide access to the set of storage switches in the FICON configuration associated with a CPC that is enabled for DPM. APIs exist to define and undefine storage switches, list storage switches, query storage switch properties, update selected properties of storage switches, and move a storage switch to another storage site or storage fabric.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 200 on page 461. The Storage Switch object does not support the operational status related properties.

Table 199. Storage Switch object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Switch object is of the form `/api/storage-switches/{storage-switch-id}` where `{storage-switch-id}` is the value of the **object-id** property of the Storage Switch object.
object-id	—	String (36)	The unique identifier for the storage switch instance.
class	—	String (14)	The class of the Storage Switch object is **"storage-switch"**.
parent	—	String/ URI	The parent of a storage site is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
name	(w)(pc)	String (1-64)	The display name specified for the storage switch. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage sites within the FICON configuration of its containing storage site and storage fabric. Default value: Currently of the form "Storage switch *{domain-id}*", where *{domain-id}* is the value of the **domain-id** property. This form is subject to change in the future.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage switch. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 200. Storage Switch object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
domain-id	(w)(pc)	String (2)	A two-character lowercase hexadecimal number in the range 01-ef that represents the domain identifier assigned to the storage switch. Domain identifiers must be unique to the other storage switches within the same fabric. The value may never be set to **null**, but can be **null** when queried if the switch was imported from an incomplete IOCDS.
port-count	(w)(pc)	Integer (2-256)	The number of ports on the switch. Default value: 256
storage-fabric-uri	(pc)	String/ URI	The canonical URI of the storage fabric associated with this storage switch. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage fabric to another through the `Move Storage Switch to Storage Fabric` operation
storage-site-uri	(pc)	String/ URI	The canonical URI of the storage site associated with this storage switch. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage site to another through the `Move Storage Switch to Storage Site` operation.

List Storage Switches of a Storage Site

The `List Storage Switches of a Storage Site` operation lists the storage switches associated with the storage site with the given identifier.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}/storage-switches
```

In this request, the URI variable *{storage-site-id}* is the **object-id** of the Storage Site object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`domain-id`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **domain-id** property.
`storage-fabric-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **storage-fabric-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-switches`	Array of storage-switch-info objects	Array of storage-switch-info objects, described in the next table. The returned array may be empty.

Each nested storage-switch-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/URI	Canonical URI path (**object-uri**) of the Storage Switch object.
`name`	String	The **name** property of the Storage Switch object.
`domain-id`	String	The **domain-id** property of the Storage Switch object.
`storage-fabric-uri`	String/URI	The **storage-fabric-uri** property of the Storage Switch object.

Description

This operation lists the storage switches that are associated with the identified storage site. The object URI, name, domain ID and associated storage fabric are provided for each.

If the object ID *{storage-site-id}* does not identify a Storage Site object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **domain-id** query parameters are specified, the returned list is limited to those storage switches that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

If the **storage-fabric-uri** query parameter is specified, the returned list is limited to those storage switches that have a matching **storage-fabric-uri** property. If the **storage-fabric-uri** parameter is omitted, this filtering is not done.

A storage switch is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage site is associated with a storage switch but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage switches are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 462.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 201. List Storage Switches of a Storage Site: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage site with the object ID *{storage-site-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492/storage-switches
  HTTP/1.1
x-api-session: 158gv9ldjh8x0yarl3qjx5o7xhemmtku20ejfi92rtk8hb3yln
```

Figure 214. List Storage Switches of a Storage Site: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:39:12 GMT
content-type: application/json;charset=UTF-8
content-length: 424
{
   "storage-switches":[
      {
         "domain-id":"11",
         "name":"Storage switch 11",
         "object-uri":"/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
         "storage-fabric-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
      },
      {
         "domain-id":"21",
         "name":"Storage switch 21",
         "object-uri":"/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
         "storage-fabric-uri":"/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492"
      }
   ]
}
```

Figure 215. List Storage Switches of a Storage Site: Response

List Storage Switches of a Storage Fabric

The `List Storage Switches of a Storage Fabric` operation lists the storage switches associated with the storage fabric with the given identifier.

HTTP method and URI

```
GET /api/storage-fabrics/{storage-fabric-id}/storage-switches
```

In this request, the URI variable *{storage-fabric-id}* is the **object-id** of the Storage Fabric object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`domain-id`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **domain-id** property.
`storage-site-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **storage-site-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-switches`	Array of storage-switch-info objects	Array of storage-switch-info objects, described in the next table. Returned array may be empty.

Each nested storage-switch-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Storage Switch object.
`name`	String	The **name** property of the Storage Switch object.
`domain-id`	String	The **domain-id** property of the Storage Switch object.
`storage-site-uri`	String / URI	The **storage-site-uri** property of the Storage Switch object.

Description

This operation lists the storage switches that are associated with the identified storage fabric. The object URI, name, domain ID, and associated site are provided for each.

If the object ID *{storage-fabric-id}* does not identify a Storage Fabric object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **domain-id** query parameters are specified, the returned list is limited to those storage switches that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

If the **storage-site-uri** query parameter is specified, the returned list is limited to those storage sites that have a matching **storage-site-uri** property. If the **storage-site-uri** parameter is omitted, this filtering is not done.

A storage switch is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage fabric is associated with a storage switch but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage switches are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 464.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 202. List Storage Switches of a Storage Fabric: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage fabric with the object ID *{storage-fabric-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492/storage-switches
  HTTP/1.1
x-api-session: 5iggido8gsjbbun0k77blh9bsfvdxn8swdioyonor0ohpuebfu
```

Figure 216. List Storage Switches of a Storage Fabric: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:08:30 GMT
content-type: application/json;charset=UTF-8
content-length: 416
{
   "storage-switches":[
      {
         "domain-id":"11",
         "name":"Storage switch 11",
         "object-uri":"/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
         "storage-site-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
      },
      {
         "domain-id":"10",
         "name":"Storage switch 10",
         "object-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
         "storage-site-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
      }
   ]
}
```

Figure 217. List Storage Switches of a Storage Fabric: Response

Define Storage Switch

The `Define Storage Switch` operation defines a new Storage Switch object.

HTTP method and URI

```
POST /api/console/operations/define-storage-switch
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Optional	The value to be set as the storage switch's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the storage switch's **description** property.
`domain-id`	String (2)	Required	The value to be set as the storage switch's **domain-id** property.
`port-count`	Integer (2-256)	Optional	The value to be set as the storage switch's **port-count** property.
`storage-fabric-uri`	String/ URI	Required	The value to be set as the storage switch's **storage-fabric-uri** property.
`storage-site-uri`	String/ URI	Required	The value to be set as the storage switch's **storage-site-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the newly defined Storage Switch object.

Description

This operation defines a storage switch with the values specified and then returns its **object-uri** in the response body. An Inventory Change notification and Property Change notifications for the associated storage site and storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the URI specified in the **storage-site-uri** field does not identify a storage site object on the HMC, or the URI specified in the **storage-fabric-uri** field does not identify a storage fabric on the HMC, a 404 (Not Found) status code is returned. If the storage fabric identified by *{storage-fabric-id}* already contains a storage switch with the specified **domain-id**, or if the **storage-site-uri** field references a storage site that does not reside in the same CPC as the storage fabric referenced by the **storage-fabric-uri** field, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage switch with the specified name exists within the same FICON configuration in which the containing storage size and storage fabric exist.

If the request body contents are valid, the storage switch is defined and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 466.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 203. Define Storage Switch: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage switch with the **name** specified in the request body already exists within the same FICON configuration in which the containing storage site and storage fabric exist.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A storage site with the URI specified in the **storage-site-uri** field does not exist on the HMC.
	440	A storage fabric with the URI specified in the **storage-fabric-uri** field does not exist on the HMC.
409 (Conflict)	441	The storage site referenced by the **storage-site-uri** field and the storage fabric referenced by the **storage-fabric-uri** field do not reside in the same CPC.
	445	A storage switch with the **domain-id** specified in the request body already exists within the storage fabric identified by the **storage-fabric-uri** field in the request body.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/define-storage-switch HTTP/1.1
x-api-session: 27u5v0reohtwg2ncuw2xebonfrtxpkbdst3cmc5w12jemo6iet
content-type: application/json
content-length: 181
{
   "domain-id":"10",
   "storage-fabric-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
   "storage-site-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 218. Define Storage Switch: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:31:45 GMT
content-type: application/json;charset=UTF-8
content-length: 75
{
   "object-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
}
```

Figure 219. Define Storage Switch: Response

Usage notes

Each CPC maintains its own view of the physical storage switches. When a list operation is targeted at a console, there will likely be multiple storage switch instances returned that represent the same physical switch – one for each CPC to which that physical switch is configured. There is no intrinsic storage switch property that can be used to correlate storage switch instances that represent the same physical switch. It is therefore recommended that API clients adopt a naming convention that ensures storage switches that represent the same physical switch have the same value of their respective **name** and **domain-id** properties.

Undefine Storage Switch

The `Undefine Storage Switch` operation removes a storage switch definition.

HTTP method and URI

```
POST /api/storage-switches/{storage-switch-id}/operations/undefine
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch that is to be undefined.

Description

This operation removes a storage switch definition. If the storage switch contains switch ports, they will be removed as well. An Inventory Change notification and Property Change notifications for the associated storage site's and storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a storage switch object on the HMC.

If any physical connections from adapters or storage subsystems to the storage switch exist, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage switch definition is removed.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 204. Undefine Storage Switch: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage switch with the object ID *{storage-switch-id}* does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the object ID *{storage-switch-id}* was busy and the request timed out.
	446	An endpoint connection in an adapter or storage subsystem references the Storage Switch object with the object ID *{storage-switch-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492/operations/undefine
  HTTP/1.1
x-api-session: ynux0a1n9betzkcvy6evkzs7aektq3ovdtj7edbanadpobvnn
content-type: application/json
```

Figure 220. Undefine Storage Switch: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 19:14:05 GMT

<No response body>
```

Figure 221. Undefine Storage Switch: Response

Get Storage Switch Properties

The `Get Storage Switch Properties` operation retrieves the properties of a single Storage Switch object.

HTTP method and URI

```
GET /api/storage-switches/{storage-switch-id}
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Switch object as defined in the "Data model" on page 460. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Switch object as defined in the "Data model" on page 460.

If the API user does not have action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-switch-id}* does not identify a storage switch object on the HMC, a 404 (Not Found) status code is returned. [Updated by feature **dpm-smcd-partition-link-management**]

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks. [Updated by feature **dpm-smcd-partition-link-management**]

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 470.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 205. Get Storage Switch Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer**, **Configure Storage – Storage Administrator** , **Create Partition Link**, or **Partition Link Details** tasks. [Added by feature **dpm-smcd-partition-link-management**]
404 (Not Found)	1	A storage switch with **object-id** *{storage-switch-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 5067e2d7z01ujz4o5j106jy255f01248enx0p5wlnr8ti98xld
```

Figure 222. Get Storage Switch Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:55:23 GMT
content-type: application/json;charset=UTF-8
content-length: 411
{
   "class":"storage-switch",
   "description":"",
   "domain-id":"10",
   "name":"Storage switch 10",
   "object-id":"90204662-9437-11e8-9c43-fa163e27d492",
   "object-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
   "parent":"/api/console",
   "port-count":256,
   "storage-fabric-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492",
   "storage-site-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 223. Get Storage Switch Properties: Response

Update Storage Switch Properties

The `Update Storage Switch Properties` operation updates one or more of the writable properties of a storage switch.

HTTP method and URI

```
POST /api/storage-switches/{storage-switch-id}
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the Storage Switch object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 460. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage switch's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a Storage Switch object on the HMC. If the containing storage fabric already contains a storage switch with the specified **domain-id** or the **port-count** field value does not allow for the currently configured ports, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage switch with the specified name exists within the same FICON configuration in which the containing storage site and storage fabric exist.

If the request body contents are valid, the storage switch's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 206. Update Storage Switch Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage switch with the **name** specified in the request body already exists within the FICON configuration in which the containing storage site and storage fabric exist.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage switch with the **object-id** *{storage-switch-id}* does not exist on the HMC.

Table 206. Update Storage Switch Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The storage switch object with the **object-id** *{storage-switch-id}* was busy and the request timed out.
	441	The value of the **port-count** field does not allow for the currently configured ports on the Storage Switch object with the **object-id** *{storage-switch-id}*.
	445	A storage switch with the **domain-id** specified in the request body already exists within its associated storage fabric.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: z73khdzsrzllklgo1gbn095ot7bgckt4g6ki1mwggeizean3
content-type: application/json
content-length: 50
{
   "description":"Switch 10 in New York, Fabric A"
}
```

Figure 224. Update Storage Switch Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:56:44 GMT

<No response body>
```

Figure 225. Update Storage Switch Properties: Response

Move Storage Switch to Storage Site

The `Move Storage Switch to Storage Site` operation moves a storage switch from its current storage site to a different storage site.

HTTP method and URI

```
POST /api/storage-switches/{storage-switch-id}/operations/move-storage-site
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`storage-site-uri`	String/ URI	Required	The canonical URI path for the storage site to which this switch is to be moved.

Description

This operation moves a storage switch from the storage site to which it is currently associated to a different storage site within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a Storage Switch object on the HMC, or the URI specified in the **storage-site-uri** field does not identify a Storage Site object on the HMC. If the storage site specified in the **storage-site-uri** field already contains the Storage Switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage site is associated with a different FICON configuration.

If the request body contents are valid, the **storage-site-uri** property of the storage switch identified by *{storage-switch-id}* is updated to the value specified in the **storage-site-uri** field. The storage switch is removed from the **storage-switch-uris** array property of the original storage site and added to the **storage-switch-uris** property of the new storage site.

Property-change notifications on the switch's **storage-site-uri** property and on both the original and new storage site's **storage-switch-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 207. Move Storage Switch to Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	441	The storage site identified by the **storage-site-uri** field is associated with a different FICON configuration than the current storage site.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage switch with the **object-id** *{storage-switch-id}* does not exist on the HMC.
	2	The storage site identified by the **storage-site-uri** field does not exist on the HMC.

Table 207. Move Storage Switch to Storage Site: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The storage switch object with the **object-id** *{storage-switch-id}* was busy and the request timed out.
	450	The storage site identified by the **storage-site-uri** field already contains a storage switch with the **object-id** *{storage-switch-id}*.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492/operations/move-
  storage-site HTTP/1.1
x-api-session: 45bnirh6wcq2qs1a9fqyxnozbufxfv0yzjnnlq0ojtfxgs35mm
content-type: application/json
content-length: 79
{
   "storage-site-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 226. Move Storage Switch to Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:43:09 GMT

<No response body>
```

Figure 227. Move Storage Switch to Storage Site: Response

Move Storage Switch to Storage Fabric

The `Move Storage Switch to Storage Fabric` operation moves a storage switch from its current storage fabric to a different storage fabric.

HTTP method and URI

```
POST /api/storage-switches/{storage-switch-id}/operations/move-storage-fabric
```

In this request, the URI variable *{storage-switch-id}* is the object ID of the storage switch that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`storage-fabric-uri`	String/ URI	Required	The canonical URI path for the storage fabric to which this switch is to be moved.

Description

This operation moves a storage switch from the storage fabric to which it is currently associated to a different storage fabric within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-switch-id}* does not identify a Storage Switch object on the HMC, or the URI specified in the **storage-fabric-uri** field does not identify a Storage Fabric object on the HMC. If the storage fabric specified in the **storage-fabric-uri** field already contains the Storage Switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage fabric is associated with a different FICON configuration.

If the request body contents are valid, the **storage-fabric-uri** property of the storage switch identified by *{storage-switch-id}* is updated to the value specified in the **storage-fabric-uri** field. The storage switch is removed from the **storage-switch-uris** array property of the original storage fabric and added to the **storage-switch-uris** property of the new storage fabric.

Property-change notifications on the switch's **storage-fabric-uri** property and on both the original and new storage fabric's **storage-switch-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 208. Move Storage Switch to Storage Fabric: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	441	The storage fabric identified by the **storage-fabric-uri** field is associated with a different FICON configuration than the current storage fabric.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

Table 208. Move Storage Switch to Storage Fabric: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The storage switch with the **object-id** *{storage-switch-id}* does not exist on the HMC.
	2	The storage fabric identified by the **storage-fabric-uri** field does not exist on the HMC.
409 (Conflict)	2	The storage switch object with the **object-id** *{storage-switch-id}* was busy and the request timed out.
	450	The storage fabric identified by the **storage-fabric-uri** field already contains a storage switch with the **object-id** *{storage-switch-id}*.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492/operations/move-storage-fabric
   HTTP/1.1
x-api-session: 3u4lvmbq452l5fjupxm3jjuzua6i1sbqhb7fguq1l3zbha7frp
content-type: application/json
content-length: 83
{
   "storage-fabric-uri":"/api/storage-fabrics/08ad557c-9436-11e8-9c43-fa163e27d492"
}
```

Figure 228. Move Storage Switch to Storage Fabric: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 20:45:56 GMT

<No response body>
```

Figure 229. Move Storage Switch to Storage Fabric: Response

Inventory service data

Information about the Storage Switches managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage switch objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-switch" are to be included. Information for a particular storage switch is included only if the API user has access permission to that object as described in the `Get Storage Switch Properties` operation.

For each storage switch to be included, the inventory response includes an array entry for the Storage Switch object. This entry is a JSON object with the same contents as is specified in the “Response body

contents" section for "Get Storage Switch Properties" on page 470. That is, the data provided is the same as would be provided if a Get Storage Switch Properties operation were requested targeting this object.:

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a storage switch. This object would appear as one array entry in the response array:

```
{
   "class":"storage-switch",
   "description":"",
   "domain-id":"21",
   "name":"Storage switch 21",
   "object-id":"0e261690-9438-11e8-8ffe-fa163e27d492",
   "object-uri":"/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
   "parent":"/api/console",
   "port-count":256,
   "storage-fabric-uri":"/api/storage-fabrics/24c1b2a8-9436-11e8-9c43-fa163e27d492",
   "storage-site-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 230. Storage Switch object: Sample inventory data - Response

Storage Subsystem object

A Storage Subsystem object represents a single storage subsystem in the FICON configuration associated with a DPM-enabled CPC. The Storage Subsystem object APIs provide access to the set of storage subsystems in the FICON configuration. APIs exist to define and undefine storage subsystems, list storage subsystems, query storage subsystem properties, update selected properties of storage subsystems, move a storage subsystem to a different site, and to manage a storage subsystem's endpoint connections.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 210 on page 479. The Storage Subsystem object does not support the operational status related properties.

Table 209. Storage Subsystem object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Subsystem object is of the form `/api/storage-subsystems/{storage-subsystem-id}` where *{storage-subsystem-id}* is the value of the **object-id** property of the Storage Subsystem object.
object-id	—	String (36)	The unique identifier for the storage subsystem instance.
class	—	String (17)	The class of the Storage Subsystem object is **"storage-subsystem"**.
parent	—	String/ URI	The parent of a storage subsystem is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.

Table 209. Storage Subsystem object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the storage subsystem. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage subsystems within the FICON configuration of its containing storage site.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage subsystem. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 210. Storage Subsystem object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
connection-endpoints	(c)(pc)	Array of subsystem-connection-endpoint objects	Array of subsystem-connection-endpoint objects, described in the next table. A subsystem connection endpoint describes a switch or adapter to which this storage subsystem is physically connected. There can be at most 64 connection endpoints configured to a single storage subsystem. The value of this property will change, and property change notifications will be emitted, when endpoints are added and removed through the `Add Connection Endpoint` and `Remove Connection Endpoint` operations.
storage-control-unit-uris	(c)(pc)	Array of String/ URI	The list of storage control units defined within the storage subsystem. Each element in this array is the canonical URI path of a Storage Control Unit object. The value of this property will change, and property change notifications will be emitted, when storage control units are defined or undefined through the `Define Storage Control Unit` and `Undefine Storage Control Unit` operations
storage-site-uri	(pc)	String/ URI	The canonical URI of the storage site associated with this storage subsystem. The value of this property will change, and property change notifications will be emitted, when the storage switch is moved from one storage site to another through the `Move Storage Subsystem to Storage Site` operation.

A connection endpoint defines a storage switch or adapter to which a storage subsystem is physically connected.

Table 211. subsystem-connection-endpoint nested object properties			
Name	**Qualifier**	**Type**	**Description**
endpoint-class	—	String Enum	The class of the object identified by **endpoint-uri**. Values: • **"adapter"** – The storage subsystem is directly connected to an adapter. • **"storage-switch"** – The storage subsystem is connected to a storage switch.
endpoint-uri	—	String/ URI	The canonical URI path for the Storage Switch or Adapter object to which this subsystem is connected.
port-id	—	String (2)	A two-character lowercase hexadecimal number that represents the switch port. This value will be null if **endpoint-uri** references an Adapter object.

List Storage Subsystems of a Storage Site

The `List Storage Subsystems of a Storage Site` operation lists the storage subsystems associated with the storage site with the given identifier.

HTTP method and URI

```
GET /api/storage-sites/{storage-site-id}/storage-subsystems
```

In this request, the URI variable *{storage-site-id}* is the **object-id** of the Storage Site object.

Query parameters:

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-subsystems`	Array of storage-subsystem-info objects	Array of storage-subsystem-info objects, described in the next table. The returned array may be empty.

Each nested storage-subsystem-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Storage Subsystem object.
name	String	The **name** property of the Storage Subsystem object.

Description

This operation lists the storage subsystems that are associated with the identified storage site. The object URI and name are provided for each.

If the object ID *{storage-site-id}* does not identify a storage site object on the HMC, a 404 (Not Found) status code is returned.

If the **name** query parameter is specified, the returned list is limited to those storage subsystems that have a **name** property that matches the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

A storage subsystem is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage site is associated with a storage subsystem but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage subsystems are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 480.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 212. List Storage Subsystems of a Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage site with the object ID *{storage-site-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492/storage-subsystems
  HTTP/1.1
x-api-session: 2fdcirhybjv0ogapni5r3z0oo86u7qj87id7rlcons4ce2jfiw
```

Figure 231. List Storage Subsystems of a Storage Site: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:29:44 GMT
content-type: application/json;charset=UTF-8
content-length: 216
{
   "storage-subsystems":[
      {
         "name":"DS8886 A",
         "object-uri":"/api/storage-subsystems/37af8766-943e-11e8-8ffe-fa163e27d492"
      },
      {
         "name":"DS8870 A",
         "object-uri":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
      }
   ]
}
```

Figure 232. List Storage Subsystems of a Storage Site: Response

Define Storage Subsystem

The `Define Storage Subsystem` operation defines a new Storage Subsystem object.

HTTP method and URI

```
POST /api/console/operations/define-storage-subsystem
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Required	The value to be set as the storage subsystem's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the storage subsystem's **description** property.
`storage-site-uri`	String/ URI	Required	The value to be set as the storage subsystem's **storage-site-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the newly defined Storage Subsystem object.

Description

This operation defines a storage subsystem with the specified values and then returns its **object-uri** in the response body. An Inventory Change notification and a property-change notification on the associated storage site's **storage-subsystem-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the URI specified in the **storage-site-uri** field does not identify a storage site object on the HMC, a 404 (Not

Found) status code is returned. If another storage subsystem with the specified **name** exists within the same FICON configuration in which the containing storage site exists, a 400 (Bad Request) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage subsystem is defined and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 482.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 213. Define Storage Subsystem: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage subsystem with the **name** specified in the request body already exists within the same FICON configuration in which the containing storage site exists.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A storage site with the URI specified in the **storage-site-uri** field does not exist on the HMC.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/define-storage-subsystem HTTP/1.1
x-api-session: 4mmbozhl0i5mvnrlosv8r1r5e5hrfd7gc82pkpzqbm0cv6bepx
content-type: application/json
content-length: 99
{
   "name":"DS8870 A",
   "storage-site-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 233. Define Storage Subsystem: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:21:02 GMT
content-type: application/json;charset=UTF-8
content-length: 77
{
   "object-uri":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492"
}
```

Figure 234. Define Storage Subsystem: Response

Usage notes

Each CPC maintains its own view of the physical storage subsystems. When a list operation is targeted at a console, there will likely be multiple storage subsystem instances returned that represent the same physical subsystem – one for each CPC to which that physical subsystem is configured. There is no intrinsic storage subsystem property that can be used to correlate storage subsystem instances that represent the same physical subsystems. It is therefore recommended that API clients adopt a naming convention that ensures storage subsystems that represent the same physical subsystem have the same value of their respective **name** properties.

Undefine Storage Subsystem

The `Undefine Storage Subsystem` operation removes a storage subsystem definition.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/undefine
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem that is to be undefined.

Description

This operation removes a storage subsystem definition. If the storage subsystem contains storage control units or connection endpoints, they will be removed as well. Switch or adapter objects referenced by a deleted endpoint are not deleted. Inventory Change notifications for the removed storage subsystem and for each of its contained storage control units are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC.

If physical connections from adapters or storage switches to the storage subsystem exist, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage subsystem definition, and all its storage control units and connection endpoints, are removed.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 214. Undefine Storage Subsystem: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage subsystem with the object ID *{storage-subsystem-id}* does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the object ID *{storage-subsystem-id}* was busy and the request timed out.
	446	The **connection-endpoints** property of the Storage Subsystem object with the object ID *{storage-subsystem-id}* is not empty.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  undefine HTTP/1.1
x-api-session: 67zvhvf3c5z2uu0euonieit063n2spw26aco76zkvt422iir6a
content-type: application/json
```

Figure 235. Undefine Storage Subsystem: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:20:28 GMT

<No response body>
```

Figure 236. Undefine Storage Subsystem: Response

Get Storage Subsystem Properties

The `Get Storage Subsystem Properties` operation retrieves the properties of a single Storage Subsystem object.

HTTP method and URI

```
GET /api/storage-subsystems/{storage-subsystem-id}
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the Storage Subsystem object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Subsystem object as defined in the "Data model" on page 478. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Subsystem object as defined in the "Data model" on page 478.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 486.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 215. Get Storage Subsystem Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage subsystem with **object-id** *{storage-subsystem-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 3fbndln8qdk7d0vshl0ltfg8fxxpx8zqp0emq2xbl8gnuwhrvk
```

Figure 237. Get Storage Subsystem Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:09:00 GMT
content-type: application/json;charset=UTF-8
content-length: 620
{
   "class":"storage-subsystem",
   "connection-endpoints":[
      {
         "endpoint-class":"storage-switch",
         "endpoint-uri":"/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
         "port-id":"00"
      },
      {
         "endpoint-class":"storage-switch",
         "endpoint-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
         "port-id":"00"
      }
   ],
   "description":"DS8870 in New York",
   "name":"DS8870 A",
   "object-id":"76c30590-943e-11e8-9c43-fa163e27d492",
   "object-uri":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
   "parent":"/api/console",
   "storage-control-unit-uris":[],
   "storage-site-uri":"/api/storage-sites/13ff101c-941f-11e8-a0c0-fa163e27d492"
}
```

Figure 238. Get Storage Subsystem Properties: Response

Update Storage Subsystem Properties

The `Update Storage Subsystem Properties` operation updates one or more of the writable properties of a storage subsystem.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the Storage Subsystem object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 478. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage subsystem's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a Storage Subsystem object on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another storage subsystem with the specified **name** exists within the same FICON configuration on which the containing storage site exists.

If the request body contents are valid, the storage subsystem's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 216. Update Storage Subsystem Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage subsystem with the **name** specified in the request body already exists within the FICON configuration in which the containing storage site exists.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage subsystem with the **object-id** *{storage-subsystem-id}* does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the **object-id** *{storage-subsystem-id}* was busy and the request timed out.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492 HTTP/1.1
x-api-session: 1zwn9re00ctj6qu09fxgu6y28lb8la7dm752gbe60m7uyg5z1m
content-type: application/json
content-length: 37
{
   "description":"DS8870 in New York"
}
```

Figure 239. Update Storage Subsystem Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:24:35 GMT

<No response body>
```

Figure 240. Update Storage Subsystem Properties: Response

Move Storage Subsystem to Storage Site

The `Move Storage Subsystem to Storage Site` operation moves a storage subsystem from its current storage site to a different storage site.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/move-storage-site
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem that is to be moved.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`storage-site-uri`	String/ URI	Required	The canonical URI path for the storage site to which this subsystem is to be moved.

Description

This operation moves a storage subsystem from the storage site to which it is currently associated to a different storage site within the same FICON configuration.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a Storage Subsystem object on the HMC, or the URI specified in the **storage-site-uri** field does not identify a Storage Site object on the HMC. If the storage site specified in the **storage-site-uri** field already contains the Storage Subsystem, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the new storage site is associated with a different FICON configuration.

If the request body contents are valid, the **storage-site-uri** property of the storage subsystem identified by *{storage-subsystem-id}* is updated to the value specified in the **storage-site-uri** field. The storage subsystem is removed from the **storage-subsystem-uris** array property of the original storage site and added to the **storage-subsystem-uris** property of the new storage site.

Property change notifications on the subsystem's **storage-site-uri** property and on both the original and new storage site's **storage-subsystem-uris** properties are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 217. Move Storage Subsystem to Storage Site: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	441	The storage site identified by the **storage-site-uri** field is associated with a different FICON configuration than the current storage site.

Table 217. Move Storage Subsystem to Storage Site: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage subsystem with the **object-id** *{storage-subsystem-id}* does not exist on the HMC.
	2	The storage site identified by the **storage-site-uri** field does not exist on the HMC.
409 (Conflict)	2	The storage subsystem object with the **object-id** *{storage-subsystem-id}* was busy and the request timed out.
	450	The storage site identified by the **storage-site-uri** field already contains a storage subsystem with the **object-id** *{storage-subsystem-id}*.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  move-storage-site HTTP/1.1
x-api-session: 3uyn7nr37w44fg2j59hytrtc4ogmw46e5khw1mcbi3undieovv
content-type: application/json
content-length: 79
{
   "storage-site-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```

Figure 241. Move Storage Subsystem to Storage Site: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:27:40 GMT

<No response body>
```

Figure 242. Move Storage Subsystem to Storage Site: Response

Add Connection Endpoint

The `Add Connection Endpoint` operation adds a connection endpoint to a storage subsystem.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/add-connection-endpoint
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem to which a connection endpoint is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`endpoint-uri`	String/ URI	Required	The canonical URI path for the Storage Switch or Adapter object to which this subsystem is connected.
`port-id`	String (2)	Optional	A two-character lowercase hexadecimal number that represents the switch port. This value is required if **endpoint-uri** references a Storage Switch object and is prohibited if **endpoint-uri** references an Adapter object.

Description

This operation adds a connection endpoint definition to a storage subsystem. A Property Change notification for the storage subsystem's **connection-endpoints** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, or if the endpoint URI does not identify an adapter or storage switch on the HMC.

If the specified connection endpoint already exists on the storage subsystem, or if the port specified by **port-id** does not exist on the target storage switch, or if the port specified by **port-id** is already configured as a connection endpoint of another storage subsystem, or if there are already 64 endpoints defined for the target storage subsystem, or **endpoint-uri** identifies a switch in a different storage site, or **endpoint-uri** identifies an adapter port when one or more fabrics exists, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **port-id** field is specified for an adapter endpoint.

If the request body contents are valid, the new connection endpoint is added to the **connection-endpoints** property of the storage subsystem.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 218. Add Connection Endpoint: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	15	The **port-id** field is present in the request body when the **endpoint-uri** field references an Adapter object
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage subsystem with the **object-id** *{storage-subsystem-id}* does not exist on the HMC.
	2	The adapter or storage switch referenced by the **endpoint-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage subsystem object with the **object-id** *{storage-subsystem-id}* was busy and the request timed out.
	442	The port identified by **port-id** is not defined on the endpoint switch.
	443	The specified connection endpoint already exists on the storage subsystem object with the **object-id** *{storage-subsystem-id}*.
	455	A new endpoint cannot be created to an adapter port when storage fabrics are defined.
	456	A new endpoint cannot be created to a storage switch that exists in a different storage site.
	457	The switch port identified by the **port-id** field is already configured as a connection endpoint of another storage subsystem.
	486	A new endpoint cannot be created for the storage subsystem with the **object-id** *{storage-subsystem-id}* because the maximum number of endpoints (64) for that storage subsystem are already defined.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  add-connection-endpoint HTTP/1.1
x-api-session: 4ksgl9ak7cjeah3fiqwm3arl003hdrqi7uy68bdso7g3z2iopj
content-type: application/json
content-length: 95
{
   "endpoint-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
   "port-id":"00"
}
```

Figure 243. Add Connection Endpoint: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:44:11 GMT

<No response body>
```

Figure 244. Add Connection Endpoint: Response

Remove Connection Endpoint

The `Remove Connection Endpoint` operation removes a connection endpoint from a storage subsystem.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/remove-connection-endpoint
```

In this request, the URI variable *{storage-subsystem-id}* is the object ID of the storage subsystem from which the connection endpoint is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`endpoint-uri`	String/ URI	Required	The canonical URI path for the Storage Switch or Adapter object from which this subsystem is to be disconnected.
`port-id`	String (2)	Optional	A two-character lowercase hexadecimal number that represents the switch port. This value is required if **endpoint-uri** references a Storage Switch object and is ignored if **endpoint-uri** references an Adapter object.

Description

This operation removes a connection endpoint definition from a storage subsystem. A Property Change notification for the storage subsystem's **connection-endpoints** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, or if **endpoint-uri** does not identify an adapter or storage switch on the HMC.

If the specified connection endpoint does not exist on the storage subsystem, or the connection endpoint is used by a storage path, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the specified connection endpoint is removed from the **connection-endpoints** property of the storage subsystem.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 219. Remove Connection Endpoint: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage subsystem with the **object-id** *{storage-subsystem-id}* does not exist on the HMC.
	2	The adapter or storage switch referenced by the **endpoint-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage subsystem object with the **object-id** *{storage-subsystem-id}* was busy and the request timed out.
	444	The specified connection endpoint does not exist on the storage subsystem object with the **object-id** *{storage-subsystem-id}*.
	446	The specified connection endpoint is configured in one or more of the storage paths defined in the storage subsystem object with the **object-id** *{storage-subsystem-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  remove-connection-endpoint HTTP/1.1
x-api-session: 1ern4xd5sqegdqh5tlcvwp7quls9q55uuu7ahbc8ue3ywfva0g
content-type: application/json
content-length: 95
{
   "endpoint-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
   "port-id":"00"
}
```

Figure 245. Remove Connection Endpoint: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 30 Jul 2018 21:43:47 GMT

<No response body>
```

Figure 246. Remove Connection Endpoint: Response

Inventory service data

Information about the Storage Subsystems managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage subsystem objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-subsystem" are to be included. Information for a particular storage subsystem is included only if the API user has access permission to that object as described in the `Get Storage Subsystem Properties` operation.

For each storage subsystem to be included, the inventory response array includes the following:

- An array entry for the storage subsystem object. This entry is a JSON object with the same contents as is specified in the Response body contents section for “Get Storage Subsystem Properties” on page 486. That is, the data provided is the same as would be provided if a Get Storage Subsystem Properties operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a storage subsystem. This object would appear as one array entry in the response array:

```
{
   "class":"storage-subsystem",
   "connection-endpoints":[
      {
         "endpoint-class":"storage-switch",
         "endpoint-uri":"/api/storage-switches/0e261690-9438-11e8-8ffe-fa163e27d492",
         "port-id":"00"
      },
      {
         "endpoint-class":"storage-switch",
         "endpoint-uri":"/api/storage-switches/fdc68064-9437-11e8-8ffe-fa163e27d492",
         "port-id":"00"
      }
   ],
   "description":"",
   "name":"DS8886 B",
   "object-id":"9b15669a-943e-11e8-8ffe-fa163e27d492",
   "object-uri":"/api/storage-subsystems/9b15669a-943e-11e8-8ffe-fa163e27d492",
   "parent":"/api/console",
   "storage-control-unit-uris":[],
   "storage-site-uri":"/api/storage-sites/0336a208-9434-11e8-9c43-fa163e27d492"
}
```


Figure 247. Storage Subsystem object: Sample inventory data - Response

Storage Control Unit object

A Storage Control Unit object represents a single storage control unit in the FICON configuration associated with a DPM-enabled CPC. The Storage Control Unit object APIs provide access to the set of storage control units within the FICON configuration. APIs exist to define and undefine storage control units, list storage control units, query storage control unit properties, update selected properties of storage control units, and managed volume ranges. APIs also exist to create, delete, query and update the storage path elements of a storage control unit.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 221 on page 498. The Storage Subsystem object does not support the operational status related properties.

Table 220. Storage Control Unit object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Control Unit object is of the form `/api/storage-control-units/{storage-control-unit-id}` where *{storage-control-unit-id}* is the value of the **object-id** property of the Storage Control Unit object.
object-id	—	String (36)	The unique identifier for the storage control unit instance.
class	—	String (20)	The class of the Storage Control Unit object is **"storage-control-unit"**.
parent	—	String/ URI	The parent of a storage control unit is conceptually its owning storage subsystem, and so the parent value is the canonical URI path for the subsystem.

Table 220. Storage Control Unit object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the storage control unit. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage control units within the parent storage subsystem. Default value: Currently of the form Control unit *{logical-address}*, where *{logical-address}* is the value of the **logical-address** property. This form is subject to change in the future.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the storage control unit. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 221. Storage Control Unit object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
logical-address	(w)(pc)	String (2)	A two-character lowercase hexadecimal number that represents the logical address that uniquely identifies the storage control unit within its parent storage subsystem. Logical addresses must be unique to the other storage control units within the parent storage subsystem.
storage-path-uris	(c)(pc)	Array of String/ URI	The list of paths that connect this storage control unit optionally to a port on a storage switch, and ultimately to an adapter. Each element in this array is the canonical URI path of a Storage Path element object, described Table 223 on page 499. There can be a maximum of 8 storage paths that Configure a single control unit, so the maximum number of entries in this array is 8. The value of this property will change, and property change notifications will be emitted, when storage paths are added and removed through the `Create Storage Path` and `Delete Storage Path` operations.
volume-ranges	(c)(pc)	Array of volume -range objects	Array of volume-range objects, described in Table 222 on page 499. A volume range describes a contiguous set of storage volume unit addresses that are managed by the storage control unit. The value of this property will change, and property change notifications will be emitted, when ports are added and removed through the `Add Volume Range` and `Remove Volume Range` operations.

A volume range defines a contiguous set of base or alias volumes within a storage control unit.

Table 222. Storage Control Unit object: volume-range nested object properties

Name	Qualifier	Type	Description
starting-volume	—	String (2)	A two-character lowercase hexadecimal number that represents the first unit address in the volume range.
ending-volume	—	String (2)	A two-character lowercase hexadecimal number that represents the last unit address in the volume range. Default value: The ending volume will be the same as the starting volume, thus giving a range of one volume.
type	—	String Enum	The volume type. Values: • **"base"** – The volumes in the range are base volumes. • **"alias"** – The volumes in the range are alias volumes. Default value: **"base"**

Storage Path element object

A storage path defines a communications path from a storage control unit to an adapter, optionally through one or two storage switches.

Table 223. Storage Control Unit object: storage path element object properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for the storage path element object, of the form `/api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}`, where *{storage-control-unit-id}* is the **object-id** of the containing storage control unit, and *{storage-path-id}* is the **element-id** of this storage path.
element-id	—	String (36)	The unique identifier for the storage path instance.
class	—	String (12)	The class of a storage path element object is **"storage-path"**.
parent	—	String/ URI	The parent of a storage path is its owning storage control unit, so the parent value is the canonical URI path for the storage control unit.
adapter-port-id	(w)(pc)	String/ URI	The canonical URI path for the storage adapter port at the opposite end of the storage path.
exit-switch-uri	(w)(pc)	String/ URI	The canonical URI path for the exit switch to which the control unit, and in non-cascaded switch configurations, the adapter, are connected. The value of this property may be **null**, which indicates the adapter and control unit have a point-to-point connection without any intervening switches. Default value: **null**

Table 223. Storage Control Unit object: storage path element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
exit-port	(w)(pc)	String (2)	A two-character lowercase hexadecimal number that represents the port on the exit switch to which the control unit is connected. The value of this property will be **null** when the value of **exit-switch-uri** is **null**, and non-null when the value of **exit-switch-uri** is non-null. Default value: **null**

List Storage Control Units of a Storage Subsystem

The `List Storage Control Units of a Storage Subsystem` operation lists the storage control units managed by the storage subsystem with the given identifier.

HTTP method and URI

```
GET /api/storage-subsystems/{storage-subsystem-id}/storage-control-units
```

In this request, the URI variable *{storage-subsystem-id}* is the **object-id** of the Storage Subsystem object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`logical-address`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **logical-address** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-control-units`	Array of storage-control-unit-info objects	Array of storage-control-unit-info objects, described in the next table. The returned array may be empty.

Each nested storage-control-unit-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/URI	Canonical URI path (**object-uri**) of the Storage Control Unit object.
`name`	String	The **name** property of the Storage Control Unit object.
`logical-address`	String	The **logical-address** property of the Storage Control Unit object.

Description

This operation lists the storage control units that are managed by the identified storage subsystem. The object URI, name and logical address are provided for each.

If the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned.

If the **name** or **logical-address** query parameters are specified, the returned list is limited to those storage control units that have the same-named property matching the specified filter pattern. If any parameter is omitted, this filtering on that property is not done.

A storage control unit is included in the list only if the API user has task permission for the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks. If the specified storage subsystem is the parent of a storage control unit but the API user does not have permission to it, that object is simply omitted from the list but no error status code results.

If no storage control units are to be included in the results due to filtering or lack of task permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 500.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 224. List Storage Control Units of a Storage Subsystem: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage subsystem with the object ID *{storage-subsystem-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/storage-control-units
   HTTP/1.1
x-api-session: 1eija5xbm9fj0z3bmiwbnw37ks9esycx5tkzc4lz65c6f1ezib
```

Figure 248. List Storage Control Units of a Storage Subsystem: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:55:39 GMT
content-type: application/json;charset=UTF-8
content-length: 156
{
   "storage-control-units":[
      {
         "logical-address":"50",
         "name":"Control unit 50",
         "object-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
      }
   ]
}
```

Figure 249. List Storage Control Units of a Storage Subsystem: Response

Define Storage Control Unit

The `Define Storage Control Unit` operation defines a new Storage Control Unit object within a parent storage subsystem.

HTTP method and URI

```
POST /api/storage-subsystems/{storage-subsystem-id}/operations/define-storage-control-unit
```

In this request, the URI variable *{storage-subsystem-id}* is the **object-id** of the Storage Subsystem object on which a storage control unit is to be defined.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Optional	The value to be set as the storage control unit's **name** property.
`description`	String (0-1024)	Optional	The value to be set as the storage control unit's **description** property.
`logical-address`	String (2)	Required	The value to be set as the storage control unit's **logical-address** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/URI	The **object-uri** property of the newly defined Storage Control Unit object.

Description

This operation defines a storage control unit with the values specified in the identified storage subsystem and then returns its **object-uri** in the response body. An Inventory Change notification and a Property Change notification on the parent storage subsystem's **storage-control-unit-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-subsystem-id}* does not identify a storage subsystem object on the HMC, a 404 (Not Found) status code is returned. If the storage subsystem identified by *{storage-subsystem-id}* already contains a storage control unit with the specified **name**, a 400 (Bad Request) status code is returned. If the storage subsystem identified by *{storage-subsystem-id}* already contains a storage control unit with the specified **logical-address**, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage control unit is defined and its properties are set to their corresponding request body contents field's values. If a field is not found in the request body, its property's value will be defaulted. The new storage control unit's URI is added to the parent storage subsystem's **storage-control-unit-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 502.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 225. Define Storage Control Unit: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage control unit with the **name** specified in the request body already exists within the parent storage subsystem with the **object-id** *{storage-subsystem-id}*.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage subsystem with the **object-id** *{storage-subsystem-id}* does not exist on the HMC.
409 (Conflict)	447	A storage control unit with the **logical-address** specified in the request body already exists within the parent storage subsystem with the **object-id** *{storage-subsystem-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492/operations/
  define-storage-control-unit HTTP/1.1
x-api-session: 557u9y6zn6qwei7239zy3roucdrvzkr8ubni9ijykoe6eql2tu
content-type: application/json
content-length: 25
{
   "logical-address":"50"
}
```

Figure 250. Define Storage Control Unit: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:21:54 GMT
content-type: application/json;charset=UTF-8
content-length: 80
{
   "object-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}
```

Figure 251. Define Storage Control Unit: Response

Undefine Storage Control Unit

The `Undefine Storage Control Unit` operation removes a storage control unit definition.

HTTP method and URI

`POST /api/storage-control-units/{storage-control-unit-id}/operations/undefine`

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit to be removed.

Description

This operation removes a storage control unit definition. If the storage control unit contains storage paths or volume ranges, they will be removed as well. Inventory Change notifications on the removed storage control unit and each of its contained storage paths, and a Property Change notification on the parent storage subsystem's **storage-control-unit-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC.

If the storage control unit is mapped to any storage volume, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage control unit's storage paths and volume ranges are deleted and its URI is removed from the parent storage subsystem's **storage-control-unit-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 226. Undefine Storage Control Unit: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with the object ID *{storage-control-unit-id}* does not exist on the HMC.
409 (Conflict)	2	The storage control unit with the **object-id** *{storage-control-unit}* was busy and the request timed out.
	446	The storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in one or more storage groups.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/
   undefine HTTP/1.1
x-api-session: 4er4rp2aw9y5mgokxtwb8wz37v1mx4nyyj2iqx5gct66ewhcb4
content-type: application/json
```

Figure 252. Undefine Storage Control Unit: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:20:54 GMT

<No response body>
```

Figure 253. Undefine Storage Control Unit: Response

Get Storage Control Unit Properties

The `Get Storage Control Unit Properties` operation retrieves the properties of a single Storage Control Unit object.

HTTP method and URI

```
GET /api/storage-control-units/{storage-control-unit-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Control Unit object as defined in the "Data model" on page 497. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Control Unit object as defined in the "Data model" on page 497.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 506.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 227. Get Storage Control Unit Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with **object-id** *{storage-control-unit-id}* does not exist on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492 HTTP/1.1
x-api-session: 1skejiyvyiew4b70zzxk6y2s802fh4gpugbkvy4yddsambc5c8
```

Figure 254. Get Storage Control Unit Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:51:56 GMT
content-type: application/json;charset=UTF-8
content-length: 946
{
   "class":"storage-control-unit",
   "description":"LCU 50 in DS8870 A",
   "logical-address":"50",
   "name":"Control unit 50",
   "object-id":"69bf384a-94d5-11e8-8ffe-fa163e27d492",
   "object-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
   "parent":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
   "storage-path-uris":[
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
         7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
         7f4eab9c-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
         7f4f02d6-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
         8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
   ],
   "volume-ranges":[
      {
         "ending-volume":"04",
         "starting-volume":"00",
         "type":"base"
      },
      {
         "ending-volume":"ff",
         "starting-volume":"fc",
         "type":"alias"
      }
   ]
}
```

Figure 255. Get Storage Control Unit Properties: Response

Update Storage Control Unit Properties

The `Update Storage Control Unit Properties` operation updates one or more of the writable properties of a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 497. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage control unit's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, or if the specified **logical-address** already exists within the parent storage subsystem, or if the **logical-address** field is present when target storage control unit is mapped to a storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the specified name already exists within the parent storage subsystem.

If the request body contents are valid, the storage subsystem's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 228. Update Storage Control Unit Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage control unit with the **name** specified in the request body already exists within its parent storage subsystem.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.

Table 228. Update Storage Control Unit Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The Storage Control Unit object with the **object-id** *{storage-control-unit-id}* was busy and the request timed out.
	446	The **logical-address** field is present in the request body when the storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
	447	A storage control unit with the **logical-address** specified in the request body already exists within its parent storage subsystem.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492 HTTP/1.1
x-api-session: 23qo36u6e3csdzyx6uroamwcduxrt2l3hk9phe8t9n763eeszl
content-type: application/json
content-length: 37
{
   "description":"LCU 50 in DS8870 A"
}
```

Figure 256. Update Storage Control Unit Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:23:28 GMT

<No response body>
```

Figure 257. Update Storage Control Unit Properties: Response

Add Volume Range

The `Add Volume Range` operation adds a volume range to a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/operations/add-volume-range
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit to which a volume range is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`starting-volume`	String (2)	Required	The value to be set as the volume range's **starting-volume** property.
`ending-volume`	String (2)	Optional	The value to be set as the volume range's **ending-volume** property. If present, the value of the **ending-volume** field must be greater than or equal to the value of **starting-volume**.
`type`	String Enum	Optional	The value to be set as the volume range's **type** property.

Description

This operation adds a volume range to a storage control unit. A property-change notification for the storage control unit's **volume-ranges** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC.

If the storage control unit is mapped to any storage volume, or if the volume range overlaps with any volume range that already exists on the storage control unit, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may be because the value of the **ending-volume** field is less than the value of the **starting-volume** field.

If the request body contents are valid, the volume range is added to the **volume-ranges** property of the storage control unit.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 229. Add Volume Range: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	15	The value of the **ending-volume** field is less than the value of the **starting-volume** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

Table 229. Add Volume Range: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	A storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.
409 (Conflict)	2	The Storage Control Unit object with the **object-id** *{storage-control-unit-id}* was busy and the request timed out.
	446	The storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
	448	The volume range overlaps with one that already exists on the storage control unit object with the **object-id** *{storage-control-unit-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/
   add-volume-range HTTP/1.1
x-api-session: 3zu31yqjcjwa7plkqpsceibkye5gmssmmkxbf51ok5ve8kbjnw
content-type: application/json
content-length: 64
{
   "ending-volume":"04",
   "starting-volume":"00",
   "type":"base"
}
```

Figure 258. Add Volume Range: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:27:41 GMT

<No response body>
```

Figure 259. Add Volume Range: Response

Remove Volume Range

The `Remove Volume Range` operation removes a volume range from a storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/operations/remove-volume-range
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the storage control unit from which the volume range is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`starting-volume`	String (2)	Required	The **starting-volume** property value of the volume range that is to be removed.

Description

This operation removes the entire volume range with the specified starting volume from a storage control unit. A Property Change notification for the storage control unit's **volume-ranges** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC.

If the storage control unit is mapped to any storage volume, or if a volume range with the specified **starting-volume** value does not exist on the storage control unit, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the volume range with the specified **starting-volume** is removed from the **volume-ranges** property of the storage control unit.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 230. Remove Volume Range: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.

Table 230. Remove Volume Range: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The Storage Control Unit object with the **object-id** *{storage-control-unit-id}* was busy and the request timed out.
	446	The storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
	449	A volume range with the specified **starting-volume** value does not exist on the storage control unit object with the **object-id** *{storage-control-unit-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/operations/
   remove-volume-range HTTP/1.1
x-api-session: 2guwrshqajcumqql9zxk5iighh2kxxzb5tkxqnuh9hct4etk30
content-type: application/json
content-length: 25
{
   "starting-volume":"00"
}
```

Figure 260. Remove Volume Range: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:27:21 GMT

<No response body>
```

Figure 261. Remove Volume Range: Response

Create Storage Path

The `Create Storage Path` operation creates a new Storage Path element object within a parent storage control unit.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/storage-paths
```

In this request, the URI variable *{storage-control-unit-id}* is the **object-id** of the parent Storage Control Unit object of the new storage path.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uri`	String/URI	Required	The value to be set as the storage path's **adapter-port-uri** property.
`exit-switch-uri`	String/URI	Optional	The value to be set as the storage path's **exit-switch-uri** property.
`exit-port`	String (2)	Optional	The value to be set as the storage path's **exit-port** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/URI	The element URI of the newly created Storage Path element object.

Description

This operation creates a storage path with the values specified on the identified storage control unit and then returns its element URI in the response body. The response also includes a **Location** header that provides this URI. An Inventory Change notification and a Property Change notification on the parent storage control unit’s **storage-path-uris** property are emitted asynchronously to this operation.

A storage path defines a logical connection between a storage control unit and an adapter port that flows between elements that are physically connected. If the storage path defines a direct connection between a storage control unit and an adapter port, the **endpoint-connection-uri** property of the adapter port must reference the storage control unit's storage subsystem. If the storage path references a switch, the storage subsystem must contain an endpoint connection to that switch and an endpoint connection must exist between the adapter port and any switch that resides in the same storage fabric.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC, or if **adapter-port-uri** or **exit-switch-uri** do not identify an adapter port or storage switch on the HMC, or if the port identified by the **exit-port** field is not defined on the storage switch referenced by the **exit-switch-uri** field, a 404 (Not Found) status code is returned. If the storage control unit identified by *{storage-control-unit-id}* already contains a storage path with the same set of property values, a 400 (Bad Request) status code is returned. If the storage control unit is mapped to any storage volume, or if the maximum number of storage ports is already defined for the control unit, or the specified exit port is not defined on the specified switch, or if no physical path connects the targeted storage control unit's storage subsystem to the specified adapter port through the specified storage switch, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **exit-port-id** field is present when the **exit-switch-uri** field is omitted, or the **exit-port-id** field is omitted when the **exit-switch-uri** field is present

If the request body contents are valid, the storage path is created and its properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted. The new storage path’s URI is added to the parent storage control unit’s **storage-path-uris** list property.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the adapter whose port is identified by *{adapter-port-id}*.

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 514.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 231. Create Storage Path: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage path with the same property values as those defaulted or specified in the request body already exists within the parent storage control unit with the **object-id** *{storage-control-unit-id}*.
	442	A corequisite condition on the presence of the **exit-port-id** field with respect to the presence of the **exit-switch-uri** field has been violated.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.
	2	The adapter port referenced by **adapter-port-uri** does not exist on the HMC or the API user does not have object-access permission for it.
	441	The storage switch referenced by **exit-switch-uri** does not exist on the HMC.
	442	The port identified by **exit-port** is not defined on the storage switch referenced by **exit-switch-uri**.
409 (Conflict)	2	The storage control unit object with the **object-id** *{storage-control-unit-id}* was busy and the request timed out.
	442	The port identified by **exit-port** is not defined on the storage switch referenced by **exit-switch-uri**.
	446	The parent storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
	486	A new storage path cannot be created for the storage control unit with the **object-id** *{storage-control-unit-id}* because the maximum number of storage paths (8) for that control unit are already defined.
	497	An endpoint connection does not exist between the parent storage subsystem of the storage control unit with the **object-id** *{storage-control-unit-id}* and the storage switch identified by **exit-switch-uri** or adapter port identified by **adapter-port-uri**, or an endpoint connection does not exist between the adapter port identified by **adapter-port-id** and a storage switch in the storage fabric in which the storage switch identified by **exit-switch-uri** resides.

Table 231. Create Storage Path: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths
  HTTP/1.1
x-api-session: 5iiqgy1nfzrlgjqlbuz3s06mqr8u2e0d3g5a7xf7kih4jb2cyb
content-type: application/json
content-length: 190
{
   "adapter-port-uri":"/api/adapters/20633658-941f-11e8-8625-fa163e27d492/storage-ports
     /0",
   "exit-port":"00",
   "exit-switch-uri":"/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492"
}
```

Figure 262. Create Storage Path: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
  8746e9c8-94d8-11e8-a5c5-fa163e3fe47d
cache-control: no-cache
date: Tue, 31 Jul 2018 15:43:53 GMT
content-type: application/json;charset=UTF-8
content-length: 132
{
   "element-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/
     storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
}
```

Figure 263. Create Storage Path: Response

Delete Storage Path

The `Delete Storage Path` operation deletes a storage path.

HTTP method and URI

```
DELETE /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object and the URI variable *{storage-path-id}* is the element ID of the Storage Path element object to delete.

Description

This operation deletes a storage path. An Inventory Change notification and a Property Change notification on the parent storage control unit's **storage-path-uris** property are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404

(Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a storage control unit object on the HMC, or if the element ID *{storage-path-id}* does not identify a storage path in the storage control unit.

If the storage control unit is mapped to any storage volume, a 409 (Conflict) status code is returned.

If the request is valid, the identified storage path is deleted from the parent storage control unit's **storage-path-uris** list property.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 232. Delete Storage Path: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.
	5	A storage path with **element-id** *{storage-path-id}* does not exist for the storage control unit.
409 (Conflict)	2	The storage control unit object with the **object-id** *{storage-control-unit-id}* was busy and the request timed out.
	446	The parent storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
  8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: 6a7jbbd1iipk5mayxii3p742ujjzawofuthw3yyadlonn33gip
```

Figure 264. Delete Storage Path: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:40:41 GMT

<No response body>
```

Figure 265. Delete Storage Path: Response

Get Storage Path Properties

The `Get Storage Path Properties` operation retrieves the properties of a single Storage Path element object.

HTTP method and URI

```
GET /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object and the URI variable *{storage-path-id}* is the element ID of the Storage Path element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Path object as defined in the "Data model" on page 497. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Path element object as defined in the Table 223 on page 499.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. If the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, or if the element ID *{storage-path-id}* does not identify a storage path in the storage control unit, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 518.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 233. Get Storage Path Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A storage control unit with **object-id** *{storage-control-unit-id}* does not exist on the HMC.
	5	A storage path with **element-id** *{storage-path-id}* does not exist for the storage control unit.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
  8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: 2bgudgv3iuzlc6ny30fs9v3pb0w43snsewp9dtt5h0qls29dpt
```

Figure 266. Get Storage Path Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:44:39 GMT
content-type: application/json;charset=UTF-8
content-length: 466
{
   "adapter-port-uri":"/api/adapters/20633658-941f-11e8-8625-fa163e27d492/storage-ports/
     0",
   "class":"storage-path",
   "element-id":"8746e9c8-94d8-11e8-a5c5-fa163e3fe47d",
   "element-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/
     storage-paths/8746e9c8-94d8-11e8-a5c5-fa163e3fe47d",
   "exit-port":"00",
   "exit-switch-uri":"/api/storage-switches/b65d1aee-9437-11e8-9c43-fa163e27d492",
   "parent":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}
```

Figure 267. Get Storage Path Properties: Response

Update Storage Path Properties

The `Update Storage Path Properties` operation updates one or more of the writable properties of a storage path.

HTTP method and URI

```
POST /api/storage-control-units/{storage-control-unit-id}/storage-paths/{storage-path-id}
```

In this request, the URI variable *{storage-control-unit-id}* is the object ID of the Storage Control Unit object and the URI variable *{storage-path-id}* is the element ID of the Storage Path element object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 497. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a storage path's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-control-unit-id}* does not identify a Storage Control Unit object on the HMC, or if the element ID *{storage-path-id}* does not identify a storage path in the storage control unit, or if **adapter-port-uri** or **exit-switch-uri** do not identify an adapter port or storage switch on the HMC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **exit-port-id** field is present when the **exit-switch-uri** field is omitted and the current value of the **exit-switch-uri** property is **null**, or the **exit-port-id** field is omitted and their current property values are **null** when the **exit-switch-uri** field is present, or because the updated storage path properties would be identical to an existing storage port in the parent storage control unit.

If the storage control unit is mapped to any storage volume, or the specified exit switch or port do not exist a 409 (Conflict) status code is returned.

If the request body contents are valid, the storage path's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
- Object-access permission to the adapter whose port is identified by *{adapter-port-id}*. This requirement only applies when updating the **adapter-port-id** property.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 234. Update Storage Path Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The update would result in a storage path with the same property values as a storage path that already exists within the parent storage control unit with the **object-id** *{storage-control-unit-id}*.
	442	The update would put the storage port object into a state that would violate a corequisite condition on the presence of the **exit-port-id** property with respect to the presence of the **exit-switch-uri** property.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	The storage control unit with the **object-id** *{storage-control-unit-id}* does not exist on the HMC.
	2	The adapter port referenced by **adapter-port-uri** does not exist on the HMC or the API user does not have object-access permission for it.
	5	A storage path with **element-id** *{storage-path-id}* does not exist for the storage control unit on HMC.
	441	The storage switch identified by the **exit-switch-uri** field does not exist on the HMC.
	442	The port identified by **exit-port** is not defined on the storage switch referenced by **exit-switch-uri**.
409 (Conflict)	2	The Storage Control Unit object with the **object-id** *{storage-control-id}* was busy and the request timed out.
	446	The parent storage control unit with the **object-id** *{storage-control-unit-id}* is mapped to one or more volumes in a storage group.
	497	The new values specified in the **adapter-port-uri** or **exit-switch-uri** fields would put the storage path into a state where it would not flow over a physical path defined by the endpoint connections between the storage subsystem and the referenced exit switch and adapter port.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
  8746e9c8-94d8-11e8-a5c5-fa163e3fe47d HTTP/1.1
x-api-session: bucvsrejregx2gv3p5l9rc5ylsupn4mr2o49vtlbuxkp3c3d6
content-type: application/json
content-length: 81
{
   "exit-switch-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492"
}
```

Figure 268. Update Storage Path Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:48:26 GMT

<No response body>
```

Figure 269. Update Storage Path Properties: Response

Inventory service data

Information about the Storage Control Units managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage control unit objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-control-unit" are to be included. Information for a particular storage control unit is included only if the API user has access permission to that object as described in the `Get Storage Control Unit Properties` operation.

For each storage control unit to be included, the inventory response array includes the following:

- An array entry for the storage control unit object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for “Get Storage Control Unit Properties” on page 506. That is, the data provided is the same as would be provided if a `Get Storage Control Unit Properties` operation were requested targeting this object.
- An array entry for each storage path associated with the storage control unit. For each such storage path, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for “Get Storage Path Properties” on page 518.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage control unit. These objects would appear as multiple array entries in the response array:

```
{
   "class":"storage-control-unit",
   "description":"LCU 50 in DS8870 A",
   "logical-address":"50",
   "name":"Control unit 50",
   "object-id":"69bf384a-94d5-11e8-8ffe-fa163e27d492",
   "object-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
   "parent":"/api/storage-subsystems/76c30590-943e-11e8-9c43-fa163e27d492",
   "storage-path-uris":[
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
        7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
        7f4eab9c-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
        7f4f02d6-94d9-11e8-917c-fa163e3fe47d",
      "/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/storage-paths/
        8746e9c8-94d8-11e8-a5c5-fa163e3fe47d"
   ],
   "volume-ranges":[
      {
         "ending-volume":"04",
         "starting-volume":"00",
         "type":"base"
      },
      {
         "ending-volume":"ff",
         "starting-volume":"fc",
         "type":"alias"
      }
   ]
},
{
   "adapter-port-uri":"/api/adapters/1699fc1a-941f-11e8-8625-fa163e27d492/storage-ports/0",
   "class":"storage-path",
   "element-id":"7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
   "element-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492/
     storage-paths/7f4dc8d0-94d9-11e8-917c-fa163e3fe47d",
   "exit-port":"00",
   "exit-switch-uri":"/api/storage-switches/90204662-9437-11e8-9c43-fa163e27d492",
   "parent":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492"
}
```


Figure 270. Storage Control Unit object: Sample inventory data - Response

Storage Group object

A Storage Group object represents a single storage group associated with a DPM-enabled CPC. Storage groups define a set of FCP, FICON or NVMe storage volume resources that can be attached to partitions. Storage group properties that are common to the group are represented by the Storage Group object. Each storage volume in the group is represented by the Storage Volume element object. When an FCP storage group is attached to a partition, the group's fulfilled resources are virtualized and the partition's view of them is represented by a set of Virtual Storage Resource element objects. The Storage Group object APIs provide access to the set of storage groups that are associated with a CPC that is enabled for DPM. APIs exist to create and modify storage groups and their volumes, and delete and query storage group properties. APIs also exist to query and fulfill the storage volume elements, and to query and update selected properties of the virtual storage resource elements of storage groups.

Rather than creating FCP or FICON storage groups from scratch each time, a set of one or more storage templates can be defined and used when creating a new storage group. When an FCP storage group is created from a storage template, the properties predefined in the template are copied into the new group, including any storage template volumes. Template property values can be overridden when creating the new group or by modifying the new group after it is created. See “Storage Template object” on page 635 for more information on storage templates. Storage templates cannot be created for NVMe storage groups.

An FCP or FICON storage group transitions through a number of states in its lifecycle. It is complicated by the fact that some of its attributes cannot be realized without changes to the configuration of the Storage Area Network (SAN) in which the storage resources defined in the group reside. Creation or modification

of an FCP or FICON storage group by a system administrator requires a subsequent fulfillment action by the SAN administrator before the storage resources in that group can be used by a partition. The nature of the actions required of the SAN administrator to fulfill a storage request differ depending on whether the storage group is defining FCP versus FICON resources. Fulfillment of FCP resources can be auto-detected, whereas fulfillment of FICON resources requires the SAN administrator to explicitly map the selected ECKD volumes to the requested volumes in the storage group.

Modifications to an FCP or FICON group that require a fulfillment cycle are particularly complex because the new property values must be saved in addition to the active values, which are the ones that remain in effect if the storage group is currently active or activated before the modifications are fulfilled. To accommodate these needs, the storage group class contains two versions of each property that depends on fulfillment: a base version that represents the configured value and an active version (identified by an **"active-"** prefix to the property name), which represents the value of those properties for storage groups attached to partitions that are currently active, or when they are next activated. In a new group or volume that has not yet been fulfilled, the active property values will be **null** and the base property values will contain the values of that properties that were specified, or defaulted, during group creation. Once the resources in the group are all fulfilled, the base value will be copied into the active value. If a group or volume is modified, the new values specified in the update operation will be placed into the same-named base properties. The active values continue to represent the state of the storage in partitions that are currently active, or become active before the modifications are fulfilled.

A storage group has a **fulfillment-state** property that indicates a storage group's current fulfillment state. Storage volumes also have their own **fulfillment-state** property that indicates the fulfillment state of the individual storage volume. The table below lists important steps in a storage group's lifecycle, and their effect on the group's **fulfillment-state** and other properties, and on the partitions to which they are attached.

Lifecycle Step	Comment
A group is created	The **fulfillment-state** property of the group and each volume is set to **"pending"**. The base property values will be those specified or defaulted in the create operation, and the active property values will be **null**. An email is sent to the SAN administrator requesting fulfillment of the new group's storage resources. The storage group may be attached to a partition, but if activated, the group's storage resources will not be available to the partition.
The group is fulfilled	The **fulfillment-state** property of the group and each volume is set to **"complete"**. The base property values are copied to their corresponding active properties. The group's storage resources are now available to a partition. If the storage group is currently attached to an active partition, the resources will be dynamically made available to the partition. Note that there is no guarantee that all of a storage group's resource will be fulfilled at the same time, so it is possible for a storage group to be in a state of partial fulfillment. In that case, some of the properties will have been fulfilled, indicated by the base and active property values being equal, and some have not. The group's **fulfillment-state** property will be updated to **"complete"** only after all its resources have been fulfilled.

Lifecycle Step	Comment
The group is modified	If volumes are added, or any group or volume property for which an active property exists is modified, the group **fulfillment-state** property is set to **"pending"**. The base property values will be those specified in the update operation. The active property values will remain unchanged. New and deleted volumes URIs are added to or remove from the **storage-volume-uris** property of the group. If the storage group is attached to a partition that is currently active or subsequently activated, it will continue to see the original storage resources. The **fulfillment-state** property of any new volume, or any volume with a modified base property, is set to **"pending"**. If storage volumes are created or deleted, or any group or volume property for which an active property exists is modified, an email is sent to the SAN administrator requesting fulfillment of the new or modified resources. The fulfillment state for deleted FCP storage volumes changes to **"deleting"**. Deleted FCP volumes are removed from their parent storage group only after the system detects that the backing SAN resources have been deconfigured. Deleted FICON volumes are removed immediately and do not required action by the SAN administrator. Deleted volumes are included in the email to notify the SAN administrator that those storage resources may now be recovered.
The group's new and modified resources are fulfilled	The **fulfillment-state** property of the group and each volume is set to **"complete"**. The updated base property values are copied to the active properties. The group's new or updated storage resources are now available to a partition. If attached to an active partition, its resources will be dynamically changed.
The group is attached to a partition	If the partition is currently active, the group's storage fulfilled resources are dynamically added to the partition. A group can be attached to a partition any time after it is created.
The group is detached from a partition	If the partition is currently active, the group's storage resources are dynamically removed from the partition.
The group is deleted	The group must be detached from all partitions to which it is attached before it can be deleted. The delete of the group also deletes all its storage volumes.

An NVMe storage volume is directly associated with an NVMe storage adapter, which internally contains the storage resource in the form of a Solid State Drive (SSD). No additional fulfillment actions are required, so the **fulfillment-state** property value will be **"complete"** for the storage group and all new storage volumes when an NVMe storage group is created or is later modified to add new storage volumes.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 100, with the class-specific specializations identified in Table 236 on page 526. The Storage Group object does not support the operational status related properties.

Table 235. Storage Group object: base managed object properties specializations			
Name	**Qualifier**	**Type**	**Description of specialization**
object-uri	—	String/ URI	The canonical URI path of the Storage Group object is of the form `/api/storage-groups/{storage-group-id}` where `{storage-group-id}` is the value of the **object-id** property of the Storage Group object.
object-id	—	String (36)	The unique identifier for the storage group instance.
parent	—	String/ URI	The parent of a storage group is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (13)	The class of a Storage Group object is **"storage-group"**.
name	(w)(pc)	String (1-64)	The display name specified for the storage group. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage groups associated to the same CPC.
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the storage group. Default value: If the storage group is created from a template, the value of the template's **description** property; otherwise an empty string.

Class specific additional properties

In addition to the properties defined through included schema, this object includes the following additional class-specific properties:

Table 236. Storage Group object: class specific properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this storage group object.	All
type	—	String Enum	The type of storage resources managed by the storage group. Values: • **"fcp"** - Fibre Channel Protocol • **"fc"** - Fibre Connection • **"nvme"** - Non-volatile Memory Express Default value: If a storage template is not specified when creating a storage group, the **type** property is required; otherwise the value defaults to the value of the template's **type** property.	All

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
shared	(w)(pc)	Boolean	**true** if this storage group can be attached to more than one partition; **false** if this storage group is dedicated to a single partition. This property’s value cannot be changed from true to false if the group is attached to more than one partition. This property's value must be **"false"** when the value of the **type** property is **"nvme"** or when the value of the **max-partitions** property is **1**. Default value: If a storage template is not specified when creating a storage group, the **shared** property defaults to **true**; otherwise the value defaults to the value of the template’s **shared** property.	All

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
fulfillment-state	(pc)	String Enum	The current fulfillment state of the storage group. Values: • **"complete"** – All resources in this storage group have been fulfilled. • **"pending"** – The storage group has been created or modified , but has not yet been completely fulfilled by the SAN administrator. This fulfillment state does not apply if the **type** property value is **"nvme"**. • **"pending-with-mismatches"** – One or more of the volumes of this storage group have a **fulfillment-state** property value of **"pending-with-mismatches"**, **"configuration-error"** or **"overprovisioned"**. This fulfillment state is only applicable if the **type** property value is **"fcp"**. • **"checking-migration"** – The storage group was created automatically when the **dpm-storage-management** feature was enabled for the Defined CPC. Volumes that were configured for the virtual storage resources before the feature enablement are being identified. This fulfillment state is only applicable if the **type** property value is **"fcp"**. • **"incomplete"** – For storage groups with a **type** property value of **"fcp"**, the storage group migration is complete, but at least one of its storage volumes cannot be detected. For storage groups with a **type** property value of **"nvme"**, an SSD is defective or has been removed from one or more NVMe adapters associated with this storage group. This fulfillment state does not apply if the **type** property value is **"fc"**. Default value: **"pending"**	All

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
storage-volume-uris	(c)(pc)	Array of String/ URI	The list of storage volumes in this storage group. Each element in this array is the canonical URI path of a Storage Volume object. This value will change as storage volumes are added and removed from this storage group through the `Modify Storage Group Properties` operation. Default value: If the storage group is created from a template, a set of storage volumes based on the template volumes in the template's **storage-template-volume-uris** property; otherwise an empty array.	All
virtual-storage-resource-uris	(c)(pc)	Array of String/ URI	The list of virtual storage resources for this storage group. Each element in this array is the canonical URI path of a Virtual Storage Resource object. This value will change, and property change notification emitted, when the storage group has fulfilled resources and is attached to or detached from partitions, or when the storage group is already attached to a partition and resources are fulfilled again due to a size change. Default value: An empty array.	fcp
connectivity	(w)(pc)	Integer (1-255)	The number of adapters to utilize for this storage group. When the **type** value is **"fc"**, the maximum value for the connectivity property is the lesser of **8** or the number of ports on FICON adapters of **type "fc"**. When the **type** value is **"fcp"**, the maximum value is the lesser of **255** or the number of ports on FICON adapters of **type "fcp"**. Default value: If a storage template is not specified when creating a storage group, the **connectivity** property defaults to **2** for storage groups of **type "fcp"**; or to the maximum value for storage groups of **type "fc"**; otherwise the value defaults to the value of the template’s **connectivity** property.	fc, fcp

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
active-connectivity	(pc)	Integer (1-255)	The value of **connectivity** that applies when the group is attached to partitions that are currently active or when the partitions are next activated. If the value of the **fulfillment-state** property is **"pending"** for newly created storage groups, this property value will be **null**, indicating it is not applicable. When the **type** value is **"fc"**, the maximum value for the **active-connectivity** property is the lesser of **8** or the number of ports on FICON adapters of **type "fc"**. When the **type** value is **"fcp"**, the maximum value is the lesser of **255** or the number of ports on FICON adapters of **type "fcp"**. This value will change to the value of **connectivity**, and property change notifications emitted, when the value of the **connectivity** property decreases, or after the value of the **connectivity** property increases and the new or modified storage group is fulfilled.	fcp
max-partitions	(w)(pc)	Integer	The maximum number of partitions to which this storage group can be attached. The value of **max-partitions** cannot exceed **1** if the value of the **shared** field is **false**. The value of **max-partitions** must be greater than **1** if the value of the **shared** field is **true**. The value of **max-partitions** cannot be decreased to a value that is less that the total number of partitions to which this storage group is currently attached. The minimum value for the **max-partitions** property is 1; the maximum is the value of the CPC object's **maximum-partitions** property. Default value: If a storage template is not specified when creating a storage group, the **max-partitions** property defaults to **2**; otherwise the value defaults to the value of the template's **max-partitions** property.	fcp

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
active-max-partitions	(pc)	Integer	When the value of the **max-partitions** property changes, the limit on the number of partitions to which the partition can be attached changes immediately. However, if the value increases, the storage resources for the additional partitions require fulfillment. When the storage group is attached to more than the **active-max-partitions** number of partitions, the most recently attached partitions will not see the storage group's storage resources until that fulfillment occurs. If the value of the **fulfillment-state** property is **"pending"** for newly created storage groups, this property value will be **null**, indicating it is not applicable. This value will change to the value of **max-partitions**, and property change notifications emitted, when the value of the **max-partitions** property decreases, or after the value of the **max-partitions** property increases and the new or modified storage group is fulfilled.	fcp
candidate-adapter-port-uris	(c)(pc)	Array of String/ URI	The list of adapter ports that are candidates for use in fulfilling connections to FCP devices. Each element in this array is an instance of the canonical URI path of a storage adapter port. This value will change, and property change notifications emitted, when candidate adapter port are added and removed through the `Add Candidate Adapter Ports to an FCP Storage Group` or `Remove Candidate Adapter Ports from an FCP Storage Group` operations.	fcp

Table 236. Storage Group object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
direct-connection-count	(w)(pc)	Integer (0-1000)	The number of additional virtual storage resource connections for the host that can be directly assigned to a guest virtual machine. A value of zero indicates this feature is disabled. The **direct-connection-count** property cannot be enabled (greater than zero) when the **shared** property is **true**. Default value: If a storage template is not specified when creating a storage group, the `direct-connection-count` property defaults to `0`; otherwise the value defaults to the value of the template's `direct-connection-count` property.	fcp
unassigned-world-wide-port-names	(c)(pc)	Array of world-wide-port-name info	The list of information about the worldwide port names (WWPNs) that have been allocated to support this FCP storage group, but have not yet been assigned to a virtual storage resource. Each element in this array is an instance of a world-wide-port-name-info nested object, defined in Table 240 on page 544. This value will change, and property change notifications emitted, when the `connectivity` or `max-partitions` properties change, or when virtual storage resources are created or deleted. Default value: An empty array.	fcp

Storage Volume element object

A Storage Volume element object defines the size and usage of a single storage volume within its parent storage group.

Table 237. Storage Volume element object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
element-id	—	String (32)	Unique identifier for the storage volume instance	All
element-uri	—	String/ URI	The canonical URI path for the storage port object, of the form `/api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}`, where *{storage-group-id}* is the **object-id** of the containing storage group, and *{storage-volume-id}* is the **element-id** of this storage volume.	All

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
parent	—	String/ URI	The parent of a storage volume is conceptually its owning storage group, and so the parent value is the canonical URI path for the storage group.	All
class	—	String (14)	The class of a storage volume element is **"storage-volume"**.	All
name	(w)(pc)	String (1-64)	The name of the storage volume. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage volumes associated with the parent storage group. Default value: If the storage volume is created from a template, the value of the template volume's **name** property; otherwise a string that is currently of the form "# GiB *{usage}{index}*", where # is the **size** of the volume, *{usage}* is the **usage**, and *{index}* is a number that may or may not be present to ensure name uniqueness. This form is subject to change in the future.	All
description	(w)(pc)	String (0-100)	Arbitrary text providing additional descriptive information about the volume. Default value: If the storage volume is created from a template, the value of the template volume's **description** property. If the **type** property of the parent storage group is **"nvme"** and the adapter referenced by the **adapter-uri** property has previously been assigned to another storage volume, the **description** property defaults to the value it had for the previous storage volume. Otherwise an empty string.	All

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
fulfillment-state	(pc)	String Enum	The current fulfillment state of the storage volume. Values: • **"complete"** – The storage volume resource has been fulfilled • **"configuration-error"** – A logical unit has been sensed on some, but not all, of the WWPNs defined to the parent storage group. This fulfillment state is only applicable if the **type** property value of the parent storage group is **"fcp"**. • **"deleting"** – The storage volume has been deleted, but the SAN administrator has not yet unconfigured it logical unit. This fulfillment state is only applicable if the **type** property value of the parent storage group is **"fcp"**. • **"incomplete"** – For storage groups with a **type** property value of **"fcp"**, migration for the parent storage group is complete, but this storage volume cannot be detected. For storage groups with a **type** property value of **"nvme"**, the SSD is defective or has been removed from the NVMe adapter. This fulfillment state does not apply if the **type** property value is **"fc".** • **"overprovisioned"** -Additional logical units have been sensed beyond what is required to fulfill this volume. This fulfillment state is only applicable if the **type** property value of the parent storage group is **"fcp"**. • **"pending"** – The storage volume has been created or modified, but has not yet been fulfilled by the SAN administrator. This fulfillment state does not apply if the **type** property value is **"nvme"**. • **"pending-with-mismatches"** – For storage groups with a type property value of **"fcp"**, no logical unit with a size that exactly matches the size of this volume has been sensed. A logical unit with the closest size has been selected for this volume. For storage groups with a **type** property of **"nvme"**, the SSD in the backing adapter has been replaced with one that has a different size. This fulfillment state does not apply if the **type** property value is **"fc"**.. If the **eckd-type** property has a value of **"alias"**, the value of the **fulfillment-state** property will be **null**. Default value: **"pending"**	All

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
size	(w)(pc)	Float	The size in gibibytes (GiB) for this volume. If the value of the **eckd-type** property is **"alias"**, this property value will be **null**, indicating it is not applicable, and the property cannot be updated. For volumes in a storage group of **type "nvme"**, the size is the detected size of the SSD installed in the adapter identified by **adapter-uri**, and the property cannot be updated. For volumes in storage group of **type "fcp"**, the **size** property must be between 1.00 and 1,048,576.00. For volumes in storage groups of **type "fc"**, with a **model** value other than **"EAV"**, the **size** property is fixed at the value specified in the table below and cannot be written: Model - Maximum **size** **"1"** - 0.88 **"2"** - 1.76 **"3"** - 2.64 **"9"** - 7.92 **"27"** - 25.93 **"54"** - 51.86 For volumes with a **model** value of **"EAV"**, **size** is writable and must be between 0.88 and 212489.20. Note however that the maximum is a theoretical size that may not be supported by an operating system. The value of the **size** property cannot be reduced once a storage volume is created. For volumes in storage groups of **type "fc"**, the **size** value is related to the **cylinders** property. Specifying the size in GiB will also define the number of cylinders and vice versa. The **size** and **cylinders** properties cannot be specified together as fields in a create or modify operation. Default value: If the storage volume is created from a template, the value of the template volume's **size** property.	All

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
active-size	(pc)	Float	The value of **size** that applies when the volume is attached to partitions that are currently active or when the partitions are next activated. If the value of the **fulfillment-state** property is **"pending"** for a newly created volume, or if the value of the **eckd-type** property is **"alias"**, this property value will be **null**, indicating it is not applicable. For volumes in a storage group of **type "nvme"**, the value of **active-size** will be the same as the value of **size** unless the SSD has been replaced with one of a different size. In that case, **active-size** will show the size of the new SSD and **size** will show the size of the original SSD until the size change is accepted using the `Accept Mismatched Storage Volumes` operation. For volumes in storage groups of **type "fcp"**, the **active-size** property will be between 1.00 and 1,048,576.00 For volumes in storage groups of **type "fc"**, with a **model** value other than **"EAV"**, the **active-size** property is fixed at the value specified in the table below and cannot be written: Model - Maximum **size** **"1"** - 0.88 **"2"** - 1.76 **"3"** - 2.64 **"9"** - 7.92 **"27"** - 25.93 **"54"** - 51.86 For volumes with a **model** value of **"EAV"**, **active-size** will be between 0.88 and 212489.20. This value will change to the value of **size**, and property change notifications emitted, when the new or modified storage volume is fulfilled.	All

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
usage	(w)(pc)	String Enum	The usage of the storage volume. Values: • **"boot"** – this storage volume will contain a bootable image. • **"data"** – this storage volume will contain data. • **"not-applicable"** – the **usage** property does not apply to this storage volume. This value may never be used when creating or modifying a storage volume, but may appear as the value for this property in a response body. If the value of the **usage** property is **"not-applicable"**, the property cannot be written. Default value: If the storage volume is created from a template, the value of the template volume's **usage** property. If the **type** property of the parent storage group is **"nvme"** and the adapter referenced by the **adapter-uri** property has previously been assigned to another storage volume, the **usage** property defaults to the value it had for the previous storage volume. Otherwise, the default is **"data"**.	All
uuid	—	String (16, 32)	The sensed UUID of an FCP storage volume.	fcp
model	(w)(pc)	String Enum	The 3390 model designation for the storage. If the value of the **eckd-type** property is **"alias"**, this property value will be **null**, indicating it is not applicable, and the property cannot be written. Values: **"1"** - Model 1 **"2"** - Model 2 **"3"** - Model 3 **"9"** - Model 9 **"27"** - Model 27 **"54"** - Model 54 **"EAV"** - Extended Address Volume The value of the **model** property cannot be changed to a smaller size once a storage volume is created. Default value: If the storage volume is created from a template, the value of the template volume's **model** property; otherwise this value is required when creating storage volumes of **type "fc"**.	fc

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
active-model	(pc)	String Enum	The value of **model** that applies when the volume is attached to partitions that are currently active or when the partitions are next activated. If the value of the **fulfillment-state** property is **"pending"** for a newly created volume, or if the value of the **eckd-type** property is **"alias"**, this property value will be **null**, indicating it is not applicable. Values: **"1"** - Model 1 **"2"** - Model 2 **"3"** - Model 3 **"9"** - Model 9 **"27"** - Model 27 **"54"** - Model 54 **"EAV"** - Extended Address Volume This value will change to the value of **model**, and property change notifications emitted, when the new or modified storage volume is fulfilled.	fc

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
cylinders	(w)(pc)	Integer (1113-268434453)	The size of the volume in cylinders. If the value of the **eckd-type** property is **"alias"**, this property value will be **null**, indicating it is not applicable, and the property cannot be written. If the **model** value is other than **"EAV"**, the **cylinders** property is fixed at the value specified in the table below and cannot be written: Model - Maximum **cylinders** **"1"** - 1113 **"2"** - 2226 **"3"** - 3339 **"9"** - 10017 **"27"** - 32760 **"54"** - 65520 For volumes with a **model** value of **“EAV”**, **cylinders** is writable and has a maximum value of 268434453. Note however that this is a theoretical size that may not be supported by an operating system. The value of the **cylinders** property cannot be reduced once a storage volume is created. The **cylinders** value is related to the **size** property. Specifying the size in cylinders will also define the **size** and vice versa. The **cylinders** and **size** properties cannot be specified together as fields in a create or modify operation. Default value: If the storage volume is created from a template, the value of the template volume's **cylinders** property.	fc

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
device-number	(w)(pc)	String (4)	A four-byte lower case hexadecimal string defining the device number that is assigned by default when the storage group containing this storage volume is attached to partitions. This value may be **null**, in which case the system auto-assigns a device number. Constraint: For the partition referenced by the storage group's **partition-uri** property: If the storage group **type** is **"nvme"**, this number must be unique across the device numbers of all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition. If the storage group **type** is **"fc"**, this number must be unique across the device numbers of all instances of the objects listed in "Channel-based device numbers" on page 197 associated with the partition. Default value: If the storage volume is created from a template, the value of the template volume's **device-number** property. If the **type** property of the parent storage group is **"nvme"** and the adapter referenced by the **adapter-uri** property has previously been assigned to another storage volume, the **device-number** property defaults to the value it had for the previous storage volume. Otherwise, the default is **null**.	fc, nvme
control-unit-uri	(pc)	String/ URI	The canonical URI of the logical control unit (LCU) in which the backing ECKD volume is defined. The value of this property will be **null** if a new **"base"** volume has not yet been fulfilled. This value will change, and property change notifications emitted, when the storage volume is fulfilled through the `Fulfill FICON Storage Volume` operation. Default value: **null**.	fc
eckd-type	—	String Enum	The type of the backing ECKD volume. Values: • **"base"** - The volume is an ECKD base volume. • **"alias"** - The volume is an ECKD alias volume. Default value: **"base"**	fc

Table 237. Storage Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported storage group "type" values**
unit-address	(pc)	String (2)	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the control unit. The value of this property will be **null** if a new **"base"** volume has not yet been fulfilled. This value will change, and property change notifications emitted, when the storage volume is fulfilled through the `Fulfill FICON Storage Volume` operation. Default value: **null**	fc
paths	—	Array of partition-volume-path-info objects	Information about all paths of an FCP storage volume for each partition to which the storage group is attached. Each element in the array is a partition-volume-path-info nested object, described in the next table. This field is present only when all the following conditions are true: • The storage group is attached to one or more partitions. • The **fulfillment-state** property is **"complete"** or **"pending"**. • The **uuid** property is not null.	fcp
adapter-uri	—	String/URI	The canonical URI path of the adapter that backs this storage volume.	nvme
serial-number	—	String	The serial number of the solid-state drive (SSD) installed in the NVMe adapter identified by **adapter-uri**. This value will be the same as the **ssd-serial-number** property of that adapter object. A **null** is returned if the value of the associated CPC's **status** property is not **"operating"**, **"service-required"**, **"degraded"**, or **"exceptions"**, if the installed SSD is defective, or if the value of the **ssd-is-installed** property for that adapter is **false**.	nvme
fid	—	Integer	Function ID of the associated NVMe adapter identified by **adapter-uri**, or null if the parent storage group is not associated with an active partition.	nvme

Each partition-volume-path-info object contains the following fields:

Table 238. partition-volume-path-info nested object		
Name	**Type**	**Description**
partition-uri	String/ URI	The canonical URI path of the partition to which this virtual storage resource is attached.
device-number	String (4)	A 4-character lowercase hexadecimal string that contains the device number of the virtual resource.
target-world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains a WWPN that uniquely identifies a target port in one of the storage subsystems connected to the CPC that the storage group is associated with.
logical-unit-number	String (16)	A 16-character lowercase hexadecimal string that contains the logical unit number (LUN) representing the storage device configured in the storage controller.

Virtual Storage Resource element object

A virtual storage resource defines the virtualized view of a storage adapter as seen by a partition.

Table 239. Virtual Storage Resource element object properties			
Name	**Qualifier**	**Type**	**Description**
element-id	—	String (36)	The unique identifier for the virtual storage resource instance.
element-uri	—	String/ URI	The canonical URI path for the Virtual Storage Resource element object, of the form `/api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}`, where *{storage-group-id}* is the **object-id** of the containing storage group, and *{virtual-storage-resource-id}* is the **element-id** of this virtual storage resource.
parent	—	String/ URI	The parent of a virtual storage resource is its owning storage group, so the parent value is the canonical URI path for the storage group.
class	—	String (24)	The class of a virtual storage resource element object is **"virtual-storage-resource"**.

Table 239. Virtual Storage Resource element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
name	(w)(pc)	String (1-64)	The display name specified for the virtual storage resource. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other virtual storage resources within the parent storage group.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the virtual storage resource. Default: an empty string
partition-uri	—	String/ URI	The canonical URI path of the partition to which this virtual storage resource is attached
device-number	(w)(pc)	String (4)	Device number of the virtual storage resource. The value must be a 4-digit lower case hexadecimal. The value must be unique across the device numbers of all other virtual storage resource elements and all instances of the objects listed in "Channel-based device numbers" on page 197 of the partition identified by **partition-uri**. Default value: Auto-generated
adapter-port-uri	(w)(pc)	String/ URI	The canonical URI path of the adapter Storage Port element to which this virtual storage resource is associated. The **type** field of the parent adapter must be **"fcp"**. The value of this property will be **null** if a candidate adapter has not been discovered to back this virtual storage resource. This value may not be set to **null**. This property may only be written after the virtual resource has been fulfilled, as indicated by a non-**null** current value.
world-wide-port-name	—	String (16)	A 16-character lower case hexadecimal string that contains the World Wide Port Name of the FCP virtual storage resource.
world-wide-port-name-info	—	world-wide-port-name-info object	Information about the worldwide port name allocated to the virtual storage resource. The element is an instance of a world-wide-port-name-info nested object, as described in Table 240 on page 544.

Table 239. Virtual Storage Resource element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
degraded-reasons	(pc)	Array of String Enum	The list of reasons for the degradation of the virtual storage resource. One or more of the following: • **"adapter"** - The **status** of the backing adapter of the virtual storage resource is either **"service"** or **"exceptions"**. • **"storage-configuration"** - The virtual storage resource is not able to access some of the volumes. If the **status** of the partition that this virtual storage resource is associated with is neither **"degraded"** nor **"reservation-error"**, or the support for this property is not available on the CPC associated with the parent storage group, this list will be empty.

A world-wide-port-name-info object defines properties relating to a single worldwide port name (WWPN). Each world-wide-port-name-info object contains the following fields.

Table 240. world-wide-port-name-info object: properties		
Name	**Type**	**Description**
world-wide-port-name	String (16)	A 16-character lower case hexadecimal string that contains the world wide port name of the FCP virtual storage resource.
status	String Enum	The current status of the worldwide port name. Values: • **"validated"** - The CPC can sense all the storage volumes defined in the storage group using this WWPN. • **"not-validated"** - The CPC is unable to sense one or more of the storage volumes using this WWPN. • **"unknown"** - The WWPN status is unknown. This value will only appear in property change notifications for WWPNs that are removed from the parent storage group's **unassigned-world-wide-port-names** list property. • **"incomplete"** - The CPC is unable to sense all of the storage volumes defined in the storage group using this WWPN which were sensed and validated before.

List Storage Groups

The `List Storage Groups` operation lists the storage groups known to the target Console.

HTTP method and URI

```
GET /api/storage-groups
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`fulfillment-state`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **fulfillment-state** property. Value must be a valid storage group **fulfillment-state** property value.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property value. Value must be a valid storage group **type** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-groups`	Array of storage-group-info objects	Array of storage-group-info objects, described in the next table. The returned array may be empty.

Each nested storage-group-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Storage Group object.
`cpc-uri`	String/ URI	The **cpc-uri** property of the Storage Group object.
`name`	String	The **name** property of the Storage Group object.
`fulfillment-state`	String Enum	The **fulfillment-state** property of the Storage Group object.
`type`	String Enum	The **type** property of the Storage Group object.

Description

This operation lists the storage groups that are known by the target Console. The object URI, name, fulfillment state, type and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage groups that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

If the **fulfillment-state** or **type** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage group **fulfillment-state** or **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to

those storage groups that have a **fulfillment-state** or **type** property matching the specified value. If the **fulfillment-state** or **type** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage groups that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage group is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a storage group, that object is simply omitted from the list but no error status code results.

If no storage groups are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Storage Group object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 545.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 241. List Storage Groups: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups HTTP/1.1
x-api-session: 3or8sd53i1g0swx7g3e29tnhzdc44tss6hzg3fr65bm29h1c3p
```

Figure 271. List Storage Groups: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 15:59:10 GMT
content-type: application/json;charset=UTF-8
content-length: 420
{
   "storage-groups":[
      {
         "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
         "fulfillment-state":"pending",
         "name":"FCP Storage Group",
         "object-uri":"/api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492",
         "type":"fcp"
      },
      {
         "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
         "fulfillment-state":"complete",
         "name":"FICON Group",
         "object-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492",
         "type":"fc"
      }
   ]
}
```

Figure 272. List Storage Groups: Response

Create Storage Group

The `Create Storage Group` operation creates a new storage group object. An FCP or FICON storage group can optionally be created based on a storage template. In this case, the values of the specified template's properties are copied into the same-named properties of the new storage group. The template property values may be overridden by the presence of properties in the request body. The volumes created from the template can be overridden using the `Modify Storage Group` operation after the storage group is created. See the Storage Template Object for more information on storage templates. If a template is not specified during the create operation, the storage group's property values are initialized from the fields in the request body, or defaulted if omitted, as in a standard create operation. Once a storage group is created, it loses all association with the template from which it was created. Subsequent modifications to the template do not affect the storage group.

HTTP method and URI

```
POST /api/storage-groups
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The value to be set as the storage group’s **cpc-uri** property.
`template-uri`	String/ URI	Optional	The canonical URI of the storage template on which the new storage group is to be based. This field is not allowed if the value of the **type** field is **"nvme"**.
`name`	String (1-64)	Required	The value to be set as the storage group's **name** property.
`description`	String (1-200)	Optional	The value to be set as the storage group's **description** property.

Field name	Type	Rqd/Opt	Description
`type`	String Enum	Required if **template-uri** is null; Prohibited otherwise	The value to be set as the storage group's **type** property. The **type** field is not allowed in the request body if the **template-uri** field is present.
`shared`	Boolean	Optional	The value to be set as the storage group's **shared** property.
`connectivity`	Integer	Optional	The value to be set as the storage group's **connectivity** property. The **connectivity** field is not allowed in the request body if the **type** field has a value of **"nvme"**.
`max-partitions`	Integer	Optional	The value to be set as the storage group's **max-partitions** property. The **max-partitions** field is not allowed in the request body unless the **type** field is present and has a value of **"fcp"**, or the **type** field is omitted and the value of the **type** field in the referenced template is **"fcp"**.
`direct-connection-count`	Integer (0-1000)	Optional	The value to be set as the storage group's **direct-connection-count** property. The **direct-connection-count** field is not allowed in the request body unless the **type** field is present and has a value of **"fcp"**, or the **type** field is omitted and the value of the **type** field in the referenced template is **"fcp"**. The **direct-connection-count** field value cannot be greater than 0 if the **shared** field is omitted or set to **true**.
`storage-volumes`	Array of storage-volume-request-info nested objects	Optional	The set of volume properties for each of the storage volumes in the group. An array of one or more storage-volume-request-info nested objects, where each element defines the new property values of a storage volume that is to be created. The **operation** field of each nested object element must be set to **"create"**. For storage groups of **type "fc"**, the storage-volume-request-info nested object is defined in Table 246 on page 560. For storage groups of **type "fcp"**, the storage-volume-request-info nested object is defined in Table 247 on page 561. For storage groups of **type "nvme"**, the storage-volume-request-info nested object is defined in Table 248 on page 561.

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified through email of the new storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. The **email-to-addresses** field is not allowed in the request if the **type** field has a value of **"nvme"**. Default value: **null**. No email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the new storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be **null** when the **email-to-addresses** field is **null**. The **email-cc-addresses** field is not allowed in the request if the **type** field has a value of **"nvme"**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the new storage group resources that require fulfillment. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-to-users** field is **null**. The **email-insert** field is not allowed in the request if the **type** field has a value of **"nvme"**. Default value: **null**. An email without a special text insert will be sent.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the newly created Storage Group object.
`element-uris`	Array of String/ URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the **storage-volumes** field in the request body. If no volumes were specified when creating the storage group, the **element-uris** field will be an empty list.

Description

This operation creates a storage group with the values specified and then returns its **object-uri** and the **element-uris** of each storage volume that was created in the response body. The response also includes a Location header that provides the new storage group's URI. Inventory Change notifications for the

new storage group and for each new storage volume identified in the **storage-volumes** field are emitted asynchronously to this operation.

An FCP or FICON storage group can optionally be created based on a storage template, in which case the template's URI is given in the request body. If a template is specified, the values of its properties are copied to the same-named properties in the new storage group. Template values can be overridden by the explicit presence of a group property field in the request body. If a storage template is not specified, the new group's property values are set from the field values, or take a default value.

If the API user does not have action/task permission to the `Configure Storage - System Programmer` task, a 403 (Forbidden) status code is returned. If the **cpc-uri** or **template-uri** fields do not identify a CPC object or template object to which the API user has object-access permission, a 404 (Not Found) status code is returned. A 404 (Not Found) status code is also returned if the **storage-volumes** field is present and the **adapter-uri** field in any array element does not identify an NVMe adapter to which the API user has object-access permission. If the CPC identified by the **cpc-uri** field is already associated with a storage group with the specified name, or if two or more of the new storage volumes have the same name, or if the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM or does not have the **dpm-storage-management** feature enabled or is not active, or if restrictions on the values of the **shared** and **max-partitions** property values, or the **shared** and **direct-connection-count** property values are violated, or if no adapters are configured to the specified type, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the **template-uri** field references a storage template that does not reside in the CPC referenced by the **cpc-uri** field. For storage groups of **type "nvme"**, a 409 (Conflict) status code is also returned if the **adapter-uri** field in any **storage-volumes** element references an adapter that does not have an SSD installed or is already associated with another NVMe storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage group type, or because both of, or neither of, the **size** and **cylinders** fields are defined for a new FICON storage volume.

If the request body contents are valid, the storage group is created and its properties are set to their corresponding template property values or request body content's field's values. If the **storage-volumes** field is present, the field values in each array element are used to define a new storage volume for the group. If a template is specified and the **storage-volumes** field is not present, the volumes in the template's **storage-template-volumes** property are copied into the new storage group. If a template is not specified and a field is not found in the request body, its property's value will be defaulted. The group's and volume's active property values will be set to **null**. If at least one storage volume is being created, or the storage group being created defines a property with a corresponding active property, the create requires action by the SAN administrator and the **fulfillment-state** property of the storage group is set to **"pending"**. Otherwise, the **fulfillment-state** property of the storage group is set to **"complete"**.

If the new storage group's **fulfillment-state** is **"pending"** and the **email-to-addresses** field is present and not **null** in the request body, an email containing information about the storage group and volume resources that require fulfillment is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not **null**, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the creation of the storage group. An API client should assume that a storage group was created even though the request failed with a 409 (Conflict) status code and 491 reason code. The URI of the new Storage Group object is returned in the **error-details** field in the response body. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the `Request Storage Group Fulfillment` request.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 549.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 242. Create Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage group with the **name** specified in the request body is already associated with the CPC identified by the **cpc-uri** specified in the request body.
	15	The type field value is **"fc"** and an element of the **storage-volumes** field contains both the **size** and **cylinders** fields, or neither the **size** nor **cylinders** fields.
	18	A supplied property is not valid for a storage group's type.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
	452	The value supplied in the **device-number** field of at least two of the entries in **storage-volumes** are the same.
	453	The name field for at least two of the nested entries in the **storage-volumes** field are the same.
403 (Forbidden)	1	The API user does not have action/task permission to the `Configure Storage - System Programmer` task.
404 (Not Found)	2	The CPC identified by the **cpc-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
	444	The storage template identified by the **template-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
	446	The NVMe adapter identified by the **adapter-uri** field does not exist on the HMC or the API user does not have object-access permission for it.

Table 242. Create Storage Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	5	The CPC identified by the **cpc-uri** field is not enabled for DPM.
	8	The **max-partitions** field value conflicts with the **shared** field value.
	13	The CPC identified by the **cpc-uri** field does not support the **dpm-storage-management** feature.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	441	The storage template referenced by the **template-uri** field resides in a different CPC than the one referenced by the **cpc-uri** field.
	446	The NVMe storage adapter referenced by the **adapter-uri** field in any **storage-volumes** element is already associated with another NVMe storage volume.
	478	The **type** field is present in the request body when the **template-uri** field is present, or the **type** field is not present when the **template-uri** field is omitted.
	487	No adapters are configured for the storage protocol need to support a storage group with the specified **type**.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, a new storage group will have been created. The **error-details** field of the response body contains a created-object-info object identifying the URI of the new Storage Group object. The created-object-info object is described in the next table.
	493	The **email-to-addesses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
	495	The **direct-connection-count** field value conflicts with the shared field value.
	500	The NVMe storage adapter referenced by the **adapter-uri** field in any **storage-volumes** element does not have an SSD installed.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Table 243. created-object-info object		
Field name	**Type**	**Description**
created-object-uri	String/ URI	The URI of the new Storage Group object.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-groups HTTP/1.1
x-api-session: d3dxdh07kglx7sc3wyvefv39ur355eddoghhw190cqc72qa8g
content-type: application/json
content-length: 194
{
   "connectivity":4,
   "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
   "name":"FICON Group",
   "storage-volumes":[
      {
         "model":"1",
         "operation":"create",
         "usage":"boot"
      }
   ],
   "type":"fc"
}
```

Figure 273. Create Storage Group: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492
cache-control: no-cache
date: Thu, 02 Aug 2018 19:26:13 GMT
content-type: application/json;charset=UTF-8
content-length: 202
{
   "element-uris":[
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        ec738d80-9689-11e8-aa30-fa163e27d492"
   ],
   "object-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492"
}
```

Figure 274. Create Storage Group: Response

Delete Storage Group

The `Delete Storage Group` operation deletes a storage group.

HTTP method and URI

`POST /api/storage-groups/{storage-group-id}/operations/delete`

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group to delete.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified through email of the storage group resources that are no longer needed. These email addresses will appear in the "to:" address list in the email that is sent. The **email-to-addresses** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. No email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the storage group resources that are no longer needed. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be null when the **email-to-addresses** field is **null**. The **email-cc-addresses** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the storage group resources that are no longer needed. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-to-users** field is **null**. The **email-insert** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. An email without a special text insert will be sent.

Description

This operation deletes a storage group. The storage group must be detached from all partitions before it can be deleted. The storage group's contained storage volume elements are also deleted. Inventory Change notifications for the deleted group and volume element object are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission.

If the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned. If the storage group is still attached to any partition, or if the CPC on which this storage group resource exists is not active, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails.

If the request body contents are valid, the identified storage group, and all its storage volumes, are deleted from the CPC. If the **email-to-addresses** field is present and not **null** in the request body, an email containing information about the storage group and volume resources that may now be recovered is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not **null**, its contents will be inserted into the email body. Note that a successful completion does not imply that the emails were delivered. A failure to send the email does not rollback the deletion of the storage group. An API client should assume that the storage group was deleted even though the request failed with a 409 (Conflict) status code and 491 reason code. Errors could be encountered at an email server after the request completes, for example due to an unknown email address.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 244. Delete Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	451	The **email-insert** or **email-cc-addresses** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	481	The storage group identified by *{storage-group-id}* is still attached to at least one partition.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the storage group will have been deleted.
	493	The **email-to-addesses** field is present in the request body, but the targeted console is not configured to support the sending of emails.

Table 244. Delete Storage Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/delete
  HTTP/1.1
x-api-session: 3da1nome5qa8uu9lbikho8ftvkd4lqv1bei7xlla1zcar8cbuq
content-type: application/json
content-length: 50
{
   "email-to-addresses":[
      "SamStorage@company.com"
   ]
}
```

Figure 275. Delete Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 31 Jul 2018 15:20:54 GMT

<No response body>
```

Figure 276. Delete Storage Group: Response

Get Storage Group Properties

The `Get Storage Group Properties` operation retrieves the properties of a single Storage Group object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Group object as defined in the "Data model" on page 525. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage group object as defined in the "Data model" on page 525.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 556.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 245. Get Storage Group Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492 HTTP/1.1
x-api-session: 2zfl71u0zahnx479crjykbkccjiqxf4rstmhxi56mqgeup7kjs
```

Figure 277. Get Storage Group Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:32:02 GMT
content-type: application/json;charset=UTF-8
content-length: 1067
{
   "class":"storage-group",
   "connectivity":4,
   "cpc-uri":"/api/cpcs/e4f159ce-82a2-32a9-b8f2-de66c9b02e7e",
   "description":"A sample FICON storage group",
   "fulfillment-state":"complete",
   "name":"FICON Group",
   "object-id":"ec638c1e-9689-11e8-aa30-fa163e27d492",
   "object-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492",
   "parent":"/api/console",
   "shared":true,
   "storage-volume-uris":[
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        ec738d80-9689-11e8-aa30-fa163e27d492",
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        22de1d40-968a-11e8-a0a5-fa163e27d492",
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        70322410-968a-11e8-a0a5-fa163e27d492",
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        703e39c6-968a-11e8-a0a5-fa163e27d492",
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        704a5e40-968a-11e8-a0a5-fa163e27d492",
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        705620b8-968a-11e8-a0a5-fa163e27d492"
   ],
   "type":"fc"
}
```

Figure 278. Get Storage Group Properties: Response

Modify Storage Group Properties

The `Modify Storage Group Properties` operation updates one or more of the writable properties of a storage group.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/modify
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

Fields for properties whose values are not to be changed by this operation can and should be omitted from the request body.

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Optional	The value to be set as the storage group's **name** property.
`description`	String (0-200)	Optional	The value to be set as the storage group's **description** property.
`shared`	Boolean	Optional	The value to be set as the storage group's **shared** property.

Field name	Type	Rqd/Opt	Description
`connectivity`	Integer	Optional	The value to be set as the storage group's **connectivity** property. If the current value of the storage group's **type** property is **"fcp"**, modifying the **connectivity** property requires fulfillment action from the storage administrator. The storage group's **fulfillment-state** property will change to **"pending"**. If the **email-to-list** is also present in the request body, an email will be sent containing the new **connectivity** value.
`max-partitions`	Integer	Optional	The value to be set as the storage group's **max-partitions** property. The **max-partitions** field is not allowed in the request body unless the current value of the storage group's **type** property is **"fcp"**. Modifying the **max-partitions** property requires fulfillment action from the storage administrator. The storage group's **fulfillment-state** property will change to **"pending"**. If the **email-to-list** is also present in the request body, an email will be sent containing the new **max-partitions** value.
`direct-connection-count`	Integer (0-1000)	Optional	The value to be set as the storage group's **direct-connection-count** property. The **direct-connection-count** field is not allowed in the request body unless the current value of the storage group's **type** property is **"fcp"**. The **direct-connection-count** field value cannot be greater than 0 for shared storage groups.
`storage-volumes`	Array of storage-volume-request-info nested objects	Optional	An array of storage-volume-request-info nested objects, where each element defines the existing storage volumes that are to be deleted or the new property values of a storage volume that is to be created or modified. The storage-volume-request-info nested object is defined in the tables that follow. If not specified, the storage group's volumes remain unchanged.
`email-to-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified through email of the modified storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. The **email-to-addresses** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. No email will be sent.

Field name	Type	Rqd/Opt	Description
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the modified storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be null when the **email-to-addresses** field is **null**. The **email-cc-addresses** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the modified storage group resources that require fulfillment. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-cc-users** field is **null**. The **email-insert** field is not allowed in the request if the **type** property in the target storage group has a value of **"nvme"**. Default value: **null**. An email without a special text insert will be sent.

Each nested storage-volume-request-info object contains the following fields:

*Table 246. storage-volume-request-info nested object for **"create"** operations on **"fc"** storage volumes*

Field name	Type	Rqd/Opt	Description
`operation`	String Enum	Required	This nested object contains the property values for a new storage volume that is to be created. Value: **"create"**
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`size`	Float (0.88-212489.20)	Optional	The value to be set as the storage volume's **size** property. If the **model** is **"EAV"**, the **size** or **cylinders** fields (but not both) must be specified. If the **model** is not **"EAV"**, the **size** field may not be present in the request body.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.
`model`	String Enum	Required	The value to be set as the storage volume's **model** property.

*Table 246. storage-volume-request-info nested object for **"create"** operations on **"fc"** storage volumes (continued)*			
Field name	**Type**	**Rqd/Opt**	**Description**
`cylinders`	Integer (1113-268434453)	Optional	The value to be set as the storage volume's **cylinders** property. If the **model** is **"EAV"**, the **size** or **cylinders** fields (but not both) must be specified. If the **model** is not **"EAV"**, the **cylinders** field may not be present in the request body.
`device-number`	String (4)	Optional	The value to be set as the storage volume's **device-number** property.

*Table 247. storage-volume-request-info nested object for **"create"** operations on **"fcp"** storage volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the property values for a new storage volume that is to be created. Value: **"create"**
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`size`	Float (1.00-1048576.00)	Required	The value to be set as the storage volume's **size** property.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.

*Table 248. storage-volume-request-info nested object for **"create"** operations on **"nvme"** storage volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the property values for a new storage volume that is to be created. Value: **"create"**
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.
`device-number`	String (4)	Optional	The value to be set as the storage volume's **device-number** property.
`adapter-uri`	String/URI	Required	The canonical URI path of the NVMe Adapter object that backs this storage volume.

*Table 249. storage-volume-request-info nested object for **"modify"** operations on **"fc"** storage volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the new property values for an existing storage volume that is to be modified. Value: **"modify"**
`element-uri`	String/ URI	Required	The canonical URI path for the storage volume element object that is being updated.
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`size`	Float (0.88-212489.20)	Optional	The value to be set as the storage volume's **size** property. The **size** field is optional but may not be present in the request body if the **cylinders** field is present in the request body or if the **model** value is not **"EAV"**. Modifying the **size** property requires fulfillment action from the storage administrator. The volume and parent storage group's **fulfillment-state** properties will change to **"pending"**. If the **email-to-addresses** is also present in the request body, an email will be sent containing the new **size** value.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.
`model`	String Enum	Optional	The value to be set as the storage volume's **model** property. Modifying the **model** property requires fulfillment action from the storage administrator. The volume and parent storage group's **fulfillment-state** properties will change to **"pending"**. If the **email-to-addresses** is also present in the request body, an email will be sent containing the new **model** value.
`cylinders`	Integer (1113-268434453)	Optional	The value to be set as the storage volume's **cylinders** property. The **cylinders** field is optional but may not be preset in the request body if the **size** field is present in the request body or if the **model** value is not **"EAV"**. Modifying the **cylinders** property requires fulfillment action from the storage administrator. The volume and parent storage group's **fulfillment-state** properties will change to **"pending"**. If the **email-to-addresses** is also present in the request body, an email will be sent containing the new **cylinders** value.

*Table 249. storage-volume-request-info nested object for **"modify"** operations on **"fc"** storage volumes (continued)*			
Field name	**Type**	**Rqd/Opt**	**Description**
`device-number`	String (4)	Optional	The value to be set as the storage volume's **device-number** property.

*Table 250. storage-volume-request-info nested object for **"modify"** operations on **"fcp"** storage volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the new property values for an existing storage volume that is to be modified. Value: **"modify"**
`element-uri`	String/ URI	Required	The canonical URI path for the storage volume element object that is being updated.
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`size`	Float (1.00-1048576.00)	Optional	The value to be set as the storage volume's **size** property. Modifying the **size** property requires fulfillment action from the storage administrator. The volume and parent storage group's **fulfillment-state** properties will change to **"pending"**. If the **email-to-addresses** is also present in the request body, an email will be sent containing the new **size** value.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.

*Table 251. storage-volume-request-info nested object for **"modify"** operations on **"nvme"** storage volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the new property values for an existing storage volume that is to be modified. Value: **"modify"**
`element-uri`	String/ URI	Required	The canonical URI path for the storage volume element object that is being updated.
`name`	String (1-64)	Optional	The value to be set as the storage volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage volume's **description** property.
`usage`	String Enum	Optional	The value to be set as the storage volume's **usage** property.

*Table 251. storage-volume-request-info nested object for **"modify"** operations on **"nvme"** storage volumes (continued)*			
Field name	**Type**	**Rqd/Opt**	**Description**
`device-number`	String (4)	Optional	The value to be set as the storage volume's **device-number** property.

*Table 252. storage-volume-request-info nested object for **"delete"** operations on storage volumes of all types*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object identifies an existing storage volume that is to be deleted. Value: **"delete"**
`element-uri`	String/ URI	Required	The canonical URI path for the storage volume element object that is being deleted.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
element-uris	Array of String/ URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the **storage-volumes** field in the request body. If the **storage-volumes** field did not contain any entries with **operation** equal to **"create"**, the **element-uris** field will be an empty list.

Description

This operation updates a storage group's properties with the values specified and then returns the **element-uris** of each storage volume that was created in the response body.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission.

If the group's **fulfillment-state** property value is **"checking-migration"**, or if the CPC on which this storage group resource exists is not active, or if the change would put the storage group into a state where its **shared** and **max-partitions** property values or **shared** and **direct-connection-count** property values conflict, or if the **shared** property is **false** for a storage group that is attached to more than one partition, or if the a storage volume modification would put the volume into a state where its **model**, **size** and **cylinders** property values conflict, or the **model**, **size** or **cylinders** properties are being reduced, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if a storage volume that is assigned as a partition's boot volume is deleted, or its **usage** property is changed.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage group type, or because the parent CPC is already associated with a storage group with the specified name, or because the operation would put the storage group into a state where two or more of its storage volumes would have the same name, or because both of, or neither of, the **size** and **cylinders** fields of a FICON storage volume are defined, or because the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or because any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, the storage group's properties are updated to their corresponding request body content's field's values. Optional fields may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified. The element URIs of each new and deleted storage volume will be added to, or removed from, the storage group's **storage-volume-uris** list property. If at least one property being modified has a corresponding active property, the update requires action by the SAN administrator and the **fulfillment-state** property of the storage group is set to **"pending"**. All active property values remain unchanged.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation. This includes a Property Change notification for the **storage-volume-uris** property if the operation creates or deletes storage volumes.

If the modified storage group's **fulfillment-state** is **"pending"** and the **email-to-addresses** field is present and not **null** in the request body, an email containing information about the modified storage group and volume resources that require fulfillment is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not **null**, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the modification of the storage group. An API client should assume that the storage group was modified even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the `Request Storage Group Fulfillment` request.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 564.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 253. Modify Storage Group Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage group with the **name** specified in the request body is already associated with the CPC identified by the group's **cpc-uri** property.
	15	The storage group's **type** property value is **"fc"** and an element of the **storage-volumes** field contains both the **size** and **cylinders** fields, or an **operation** value of **"create"** and neither the **size** nor **cylinders** fields.
	18	A supplied property is not valid for a storage group's type.
	450	The storage group's **type** property value is **"fc"** and an element of the **storage-volumes** field has an **operation** value of **"delete"** and an **element-uri** value that references a storage volume with an **eckd-type** property value of **"alias"**.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
	452	The value supplied in the **device-number** field of at least one of the entries in **storage-volumes** conflicts with an existing device number for another device attached to the partition associated with this storage group.
	453	The operation would put the storage group into a state where the **name** property for at least two of its storage volumes would be the same.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	5	An element of the **storage-volumes** field has an **element-uri** value that is not a member of the storage group's **storage-volume-uris** array property value.

Table 253. Modify Storage Group Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	8	The operation would result in conflicting values for the **max-partitions** and **shared** property values.
	446	The NVMe storage adapter referenced by the **adapter-uri** field in any **storage-volumes** element is already associated with another NVMe storage volume.
	471	The value of the **fulfillment-state** property for the storage group with the **object-id** *{storage-group-id}* is **"checking-migration"**.
	475	The **max-partitions** field value is less than the number of partitions to which this storage group is currently attached, or the **shared** property is **false** when the number of partitions to which the partition is attached is more than one.
	490	The storage group's **type** property value is **"fc"** and an element of the **storage-volumes** field would put the storage volume into a state where its **model**, **size** and **cylinders** property values conflict.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the storage group will have been modified.
	493	The **email-to-addresses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
	494	The **size** or **cylinders** field value is less than the storage volume's current **size** or **cylinders**, or the **model** field value represents a size that is smaller than that of the current **model**.
	495	The operation would result in conflicting values for the **direct-connection-count** and **shared** property values.
	499	A storage volume that is assigned as a partition's boot volume is being deleted, or its **usage** or **device-number** properties are being changed.
	500	The NVMe storage adapter referenced by the **adapter-uri** field in any **storage-volumes** element does not have an SSD installed.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/modify HTTP/1.1
x-api-session: 55q0cufbqgz03k1s2bmz98nxnx2ozf4e4sqzygw7q50st3zfqe
content-type: application/json
content-length: 352
{
   "description":"A sample FICON storage group",
   "storage-volumes":[
      {
         "description":"A Model 3 FICON data volume",
         "model":"3",
         "operation":"create"
      },
      {
         "description":"A Model 1 FICON boot volume",
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492",
         "operation":"modify"
      }
   ]
}
```

Figure 279. Modify Storage Group Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:27:45 GMT
content-type: application/json;charset=UTF-8
content-length: 130
{
   "element-uris":[
      "/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/
        22de1d40-968a-11e8-a0a5-fa163e27d492"
   ]
}
```

Figure 280. Modify Storage Group Properties: Response

Resend Request

The `Resend Request` operation requests fulfillment of the FCP or FICON storage group resources that require fulfillment. This operation can be invoked on any storage group with fulfillment state **"pending"**, **"incomplete"**, or **"pending-with-mismatches"**. The request will include the current resources that require fulfillment, and will therefore reflect any subsequent modifications or partial fulfillments that have occurred since the storage group was initially created or modified.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/resend-request
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object for which a fulfillment request should be sent.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Required	A set of one or more email addresses of the people that are to be notified through email of the storage group resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the storage group resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the storage group resources that require fulfillment. The text can include HTML formatting tags. Default value: **null**. An email without a special text insert will be sent.

Description

This operation notifies through email the new, deleted or modified storage group resources that require fulfillment action by the SAN administrator. The email will reflect the current fulfillment state of the storage group. The content of the email may differ from the email that was sent as a result of a previous `Create Storage Group` or `Modify Storage Group Properties` operation in the following ways:

- If the storage group was modified since the last time it was fulfilled, the email will include the accumulation of all storage group resource changes from the previous modify requests.
- The storage group resources that were part of the previous modify request but are now fulfilled will be omitted from the email.
- Storage group resources that were deleted in a modify request may not be included in the email.

If the API user does not have action/task permission to the **Configure Storage - System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission, or identifies a storage group of **type "nvme"**.

If the current storage group does not contain any new, deleted or modified resources that require fulfillment, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example, due to an unknown email address.

If the request body fails to validate, a 400 (Bad Request) status code is returned. A 400 (Bad Request) status code is also returned if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, an email is created that describes the new, modified, or deleted storage group resources that require fulfillment. If the **email-insert** field is present in the request body, its value is inserted into the body of the email. The email is then sent to the email addresses specified in the **email-to-addresses** field and copied to the email addresses specified in the **email-cc-addresses** field.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 254. Resend Request: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the **object-id** *{storage-group-id}* has a **type** value of **"nvme"**.
409 (Conflict)	1	The storage group is not in a valid state to perform the operation (must be in one of the following fulfillment states: **"pending"**, **"incomplete"**, or **"pending-with-mismatches"**, with at least one storage volume in **"pending"** or **"deleting"** fulfillment state.
	491	An error occurred when sending the email.
	493	SMTP is not configured on HMC to send email.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/a8856fb6-6d80-11e9-b629-fa163e2274e8/operations/resend-request HTTP/1.1
x-api-session: 4gbb8y8b9o3idupz1l7xcehupxmraxg5y874slliniecvs7svx
content-type: application/json
content-length: 50
{
  "email-to-addresses":[
     "roberto.fdmming@bnkincon.com"
  ]
}
```

Figure 281. Resend Request: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 08 May 2019 08:18:00 GMT

<No response body>
```

Figure 282. Resend Request: Response

Add Candidate Adapter Ports to an FCP Storage Group

The `Add Candidate Adapter Ports to an FCP Storage Group` operation adds a list of storage adapter ports to a storage group's candidate adapter ports list.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/add-candidate-adapter-ports
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group to which the candidate adapter ports are to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uris`	Array of String/ URI	Required	A list of the adapter ports that are to be added to the storage group's candidate adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port.

Description

This operation adds a list of storage adapter ports to a storage group’s candidate adapter ports list. These adapter ports become candidates for use as backing adapters when creating virtual storage resources when the group is attached to a partition. The adapter ports should have connectivity to the Storage Area Network (SAN). This operation only applies to storage groups of **type "fcp"**. Change notification for the storage group’s **candidate-adapter-port-uris** property is emitted asynchronously to this operation.

Candidate adapter port may only be added before the CPC discovers a working communications path, indicated by a **"validated" status** on at least one of the parent storage group’s WWPNs. After that point all adapter ports in the group are automatically detected and manually adding them is no longer possible.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID {storage-group-id} does not identify a storage group to which the API user has object-access permission, or if an element of the **adapter-port-uris** array does not identify a storage adapter object to which the API user has object-access permission, or if the storage group identified by the object ID {storage-group-id} is not of **type "fcp"**.

If any adapter port in the **adapter-port-uris** list is currently a member of the storage group’s candidate adapter ports list, or if at least one WWPN in the group has been discovered, or if any adapter port in the **adapter-port-uris** list references a storage adapter port that does not reside in the target storage group's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because an adapter referenced in the **adapter-port-uris** array is not a FICON adapter.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are added to the storage group's candidate adapter ports list.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to each adapter containing the ports identified in the **adapter-port-uris** array.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 255. Add Candidate Adapter Ports to an FCP Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	An adapter referenced in the **adapter-port-uris** list has an **adapter-family** property value other than **"ficon"**.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	2	An adapter containing a port referenced by a URI in the **adapter-port-uris** field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the **object-id** *{storage-group-id}* has a type value other than **"fcp"**.
409 (Conflict)	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	441	A storage adapter port in the **adapter-port-uris** list resides in a different CPC than the targeted storage group.
	471	The **status** property of at least one WWPN listed in the **world-wide-port-names** property of the storage group identified by *{storage-group-id}* has a value of **"validated"**.
	478	A storage adapter port in the **adapter-port-uris** list is already a current member of the **candidate-adapter-port-uris** list of the storage group object with the **object-id** *{storage-group-id}*.
	483	The adapter that contains the port referenced by an element in the **adapter-port-uris** field has a **type** value other than **"fcp"**.

Table 255. Add Candidate Adapter Ports to an FCP Storage Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492/operations/add-candidate-
  adapter-ports HTTP/1.1
x-api-session: sf2z7yhhiw90fj9rqevt7z3q7up3qfp62a8lraxrstzkkqv10
content-type: application/json
content-length: 93
{
   "adapter-port-uris":[
      "/api/adapters/13eb6396-941f-11e8-8625-fa163e27d492/storage-ports/0"
   ]
}
```

Figure 283. Add Candidate Adapter Ports to an FCP Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 16:07:37 GMT

<No response body>
```

Figure 284. Add Candidate Adapter Ports to an FCP Storage Group: Response

Remove Candidate Adapter Ports from an FCP Storage Group

The `Remove Candidate Adapter Ports from an FCP Storage Group` operation removes a list of storage adapter ports from a storage group's candidate adapter ports list.

HTTP method and URI

`POST /api/storage-groups/{storage-group-id}/operations/remove-candidate-adapter-ports`

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group from which the candidate adapter port is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uris`	Array of String/ URI	Required	A list of the adapter ports that are to be removed from the storage group's candidate adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port

Description

This operation removes a list of storage adapter ports from a storage group's candidate adapter ports list. This operation only applies to storage groups of **type "fcp"**. A Property Change notification for the storage group's **candidate-adapter-port-uris** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission , or if an element of the **adapter-port-uris** array does not identify a storage adapter object to which the API user has object-access permission , or if the storage group identified by the object ID {storage-group-id} is not of **type "fcp"**.

If any adapter port in the **adapter-port-uris** list is not a current member of the storage group's candidate adapter ports list or is referenced by any of the group's virtual storage resources, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because an adapter referenced in the **adapter-port-uris** array is not a FICON adapter.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are removed from the storage group's candidate adapter ports list.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to each adapter containing the ports identified in the **adapter-port-uris** array.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 256. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	An adapter referenced in the **adapter-port-uris** list has an **adapter-family** property value other than **"ficon"**.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	2	An adapter containing a port referenced by a URI in the **adapter-port-uris** field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the **object-id** *{storage-group-id}* has a **type** value other than **"fcp"**.

Table 256. Remove Candidate Adapter Ports from an FCP Storage Group: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	479	A storage adapter port in the **adapter-port-uris** list is not a current member of the **candidate-adapter-port-uris** list of the storage group object with the **object-id** *{storage-group-id}*.
	482	A storage adapter port in the **adapter-port-uris** list is referenced by at least one element of the **virtual-storage-resource-uris** in the storage group object with the **object-id** *{storage-group-id}*.
	488	A storage adapter port referenced in the **adapter-port-uris** field has been assigned to a storage volume in the storage group object with the **object-id** *{storage-group-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/491e058c-998d-11e8-a345-fa163e27d492/operations/remove-
  candidate-adapter-ports HTTP/1.1
x-api-session: 4xtwzjxbhelnpvnnl0atkg5ut7cg6rqqnldcvvskvnj3zu8ubz
content-type: application/json
content-length: 93
{
   "adapter-port-uris":[
      "/api/adapters/13eb6396-941f-11e8-8625-fa163e27d492/storage-ports/0"
   ]
}
```

Figure 285. Remove Candidate Adapter Ports from an FCP Storage Group: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 16:07:37 GMT

<No response body>
```

Figure 286. Remove Candidate Adapter Ports from an FCP Storage Group: Response

List Storage Volumes of a Storage Group

The `List Storage Volumes of a Storage Group` operation lists the storage volumes of the storage group with the given identifier.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`fulfillment-state`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **fulfillment-state** property. Value must be a valid storage volume **fulfillment-state** property value.
`maximum-size`	Integer	Optional	Filter value to limit returned object to those that have a **size** property that is less than or equal to **maximum-size**.
`minimum-size`	Integer	Optional	Filter value to limit returned object to those that have a **size** property that is greater than or equal to **minimum-size**.
`usage`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **usage** property. Value must be a valid storage volume **usage** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-volumes`	Array of storage-volume-response-info objects	Array of storage-volume-response-info nested objects, described in the next table. The returned array may be empty.

Each nested storage-volume-response-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path (**element-uri**) of the Storage Volume element object.
`name`	String	The **name** property of the storage volume element.
`fulfillment-state`	String Enum	The **fulfillment-state** property of the storage volume element.
`size`	Integer	The **size** property of the storage volume element.
`usage`	String Enum	The **usage** property of the storage volume element.

Description

This operation lists the storage volumes that are owned by the identified storage group. The element URI, name, fulfillment state, size and usage are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned

If the **name** query parameter is specified, the returned list is limited to those storage volumes that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **fulfillment-state** or **usage** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage volume **fulfillment-state** or **usage** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage volumes that have a **fulfillment-state** or **usage** property matching the specified value. If the **fulfillment-state** or **usage** parameter is omitted, this filtering is not done for that property name.

If the **minimum-size** query parameter is specified, the returned list is limited to those storage volumes that have a **size** property that is greater than or equal to the specified value. If the **maximum-size** query parameter is specified, the returned list is limited to those storage volumes that have a **size** property that is less than or equal to the specified value. When specified together, the **minimum-size** and **maximum-size** query parameters define a size range on which the volume is filtered. If either of these query parameters are omitted, the size filter is not bounded on one end. If both of these query parameters are omitted, no filtering on the **size** property is done.

If no storage volumes are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 576.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 257. List Storage Volumes of a Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage group with the object ID *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes HTTP/1.1
x-api-session: 3xx8y87o2ajorslkb17weulfnyheic1l87kqvdksngigr1e1r8
```

Figure 287. List Storage Volumes of a Storage Group: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:34:09 GMT
content-type: application/json;charset=UTF-8
content-length: 1279
{
   "storage-volumes":[
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/704a5e40-968a-11e8-a0a5-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"Alias",
         "size":0.0,
         "usage":"not-applicable"
      },
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/70322410-968a-11e8-a0a5-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"Alias",
         "size":0.0,
         "usage":"not-applicable"
      },
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/705620b8-968a-11e8-a0a5-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"Alias",
         "size":0.0,
         "usage":"not-applicable"
      },
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/ec738d80-9689-11e8-aa30-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"0.88 GiB Boot",
         "size":0.88,
         "usage":"boot"
      },
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/22de1d40-968a-11e8-a0a5-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"2.64 GiB Data",
         "size":2.64,
         "usage":"data"
      },
      {
         "element-uri":"/api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/
           storage-volumes/703e39c6-968a-11e8-a0a5-fa163e27d492",
         "fulfillment-state":"complete",
         "name":"Alias",
         "size":0.0,
         "usage":"not-applicable"
      }
   ]
}
```

Figure 288. List Storage Volumes of a Storage Group: Response

Get Storage Volume Properties

The `Get Storage Volume Properties` operation retrieves the properties of a single Storage Volume element object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the Storage Volume element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Volume object as defined in the "Data model" on page 525. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage volume object as defined in the "Storage Volume element object" on page 532.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission or if the element ID *{storage-volume-id}* does not identify a storage volume in the storage group.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 579.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 258. Get Storage Volume Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission to it.
	5	A storage volume with **element-id** *{storage-volume-id}* does not exist in the storage group on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532/storage-volumes/f3bdfdf0-
  4bba-11e9-b920-00106f23f532 HTTP/1.1
x-api-session: 5mpfhw1mcjtir9u1t4uyryduw48bp4c3fb95b696k4vex91uh6
```

Figure 289. Get Storage Volume Properties: Request

```
200 OK
server: Hardware management console API web server/2.0
cache-control: no-cache
date: Thu ,04 Apr 2019 19:36:08 GMT
content-type:application/json;charset=UTF-8
content-length: 1186
{
   "paths":[
           {
                  "device-number":"0001",
                  "logical-unit-number":"4016401400000000",
                  "partition-uri":"/api/partitions/
                     4a0bdd08-4b1b-11e9-87db-00106f23f532",
                  "target-world-wide-port-name":"50050763070b46a6"
           },
           {
                  "device-number":"0002",
                  "logical-unit-number":"4016401400000000",
                  "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
                  "target-world-wide-port-name":"50050763070b46a6"
           },
           {
                  "device-number":"0001",
                  "logical-unit-number":"4016401400000000",
                  "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
                  "target-world-wide-port-name":"50050763070046a6"
           },
           {
                  "device-number":"0002",
                  "logical-unit-number":"4016401400000000",
                  "partition-uri":"/api/partitions/4a0bdd08-4b1b-11e9-87db-00106f23f532",
                  "target-world-wide-port-name":"50050763070046a6"
           }
   ],
   "fulfillment-state":"complete",
   "parent":"/api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532",
   "element-uri":"/api/storage-groups/f2cb8d90-4bba-11e9-b920-00106f23f532/storage-volumes/
      f3bdfdf0-4bba-11e9-b920-00106f23f532",
   "description":"",
   "element-id":"f3bdfdf0-4bba-11e9-b920-00106f23f532",
   "uuid":"6005076307FFC6A60000000000001614",
   "name":"10.00 GiB Boot",
   "usage":"boot", "active-size":10.0,
   "class":"storage-volume",
   "size":10.0
}
```

Figure 290. Get Storage Volume Properties: Response

Fulfill FICON Storage Volume

The `Fulfill FICON Storage Volume` operation maps the ECKD storage resource that has been configured by a SAN administrator to a storage volume that is part of a request for new or modified FICON storage.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/
   fulfill-ficon-storage-volume
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the storage volume element.

Request body contents

Field name	Type	Rqd/Opt	Description
`control-unit-uri`	String/URI	Required	The canonical URI of the storage control unit in which the backing ECKD volume is defined.

Field name	Type	Rqd/Opt	Description
`unit-address`	String (2)	Required	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the storage control unit.

Description

This operation provides information about the ECKD storage resource that has been selected by a SAN administrator to fulfill a base storage volume that is part of a request for new or modified FICON storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to that storage resource.

Once a FICON storage volume has been fulfilled, the storage control unit and unit that has been assigned to it cannot be changed. If the **size**, **cylinders** or **model** properties of a volume are modified, the **fulfillment-state** of the volume is changed to **"pending"**, indicating that action may be necessary by the storage administrator to adjust the size of the backing ECKD volume. The **fulfillment-state** of the volume is changed back to **"complete"** by fulfilling the volume again with the exact same **control-unit-uri** and **unit-address**, indicating that the changes have been completed.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or the element ID *{storage-volume-id}* does not identify a storage volume in the storage group, or the **control-unit-uri** in the request body does not identify a storage control unit, or the current value of the group's **type** property is not **"fc"**.

If the storage volume's **fulfillment-state** property value is not **"pending"**, or if the CPC on which this storage group resource exists is not active, or if the storage group's **connectivity** property value does not equal the number of storage paths configured in the target storage control unit, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the identified unit address is not defined on the identified control unit, or if the backing ECKD volume is an alias volume or has already being used to fulfill another storage volume, or if the identified control unit and unit address would remap a previously fulfilled storage volume, or if the backing ECKD volume is in a different CPC than the target storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage volume's **fulfillment-state** property is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. The storage volume's **control-unit-uri** and **unit-address** properties are set to the same-named request body field values. The storage volume's **size** and **model** property values are copied to the **active-size** and **active-model** properties. Change notifications for the storage volume's **fulfillment-state**, **control-unit-uri**, **unit-address**, **active-size**, and **active-model** properties are emitted asynchronously to this operation. Change notification for the parent storage group's **fulfillment-state** property may be also emitted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 259. Fulfill FICON Storage Volume: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – Storage Administrator** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	2	The storage control unit identified by **control-unit-uri** does not exist on the HMC.
	4	The storage group object with the **object-id** {storage-group-id} has a **type** value other than **"fc"**.
	5	A storage volume with **element-id** *{storage-volume-id}* does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	441	The storage control unit reference by the **control-unit-uri** field resides in a different CPC than the targeted storage volume.
	471	The current value of the **fulfillment-state** property of the storage volume identified by *{storage-volume-id}* is not **"pending"**.
	473	The storage volume identified by *{storage-volume-id}* has been previously fulfilled, and the values of the **control-unit-uri** and **unit-address** fields do not exactly match the previous values.
	484	The unit address identified by **unit-address** is not a member of a volume group on the control unit identified by **control-unit-uri**.
	485	The unit address identified by **unit-address** in the control unit identified by **control-unit-uri** has already been used to fulfill another FICON storage volume.
	486	The unit address identified by **unit-address** is a member of an alias volume group on the control unit identified by **control-unit-uri**.
	492	The **connectivity** property value of the storage volume identified by *{storage-volume-id}* does not equal the number of storage paths configured in the storage control unit identified by **control-unit-uri**.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/storage-volumes/ec738d80-
  9689-11e8-aa30-fa163e27d492/operations/fulfill-ficon-storage-volume HTTP/1.1
x-api-session: 4iep1d8dyntia1rmppcfl0pmo8yghd190u1s14fu5133vevc7u
content-type: application/json
content-length: 109
{
   "control-unit-uri":"/api/storage-control-units/69bf384a-94d5-11e8-8ffe-fa163e27d492",
   "unit-address":"00"
}
```

Figure 291. Fulfill FICON Storage Volume: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 19:29:56 GMT

<No response body>
```

Figure 292. Fulfill FICON Storage Volume: Response

Fulfill FICON Storage Volumes

The `Fulfill FICON Storage Volumes` maps the ECKD storage resources that have been configured by a SAN administrator for one or more storage volumes that are part of a request for new or modified FICON storage. This operation is similar to the `Fulfill FICON Storage Volume` operation against a storage volume element, but allows the API client to fulfill multiple volumes with a single request, thus eliminating the overhead of multiple API requests.

HTTP method and URI

`POST /api/storage-groups/{storage-group-id}/operations/fulfill-ficon-storage-volumes`

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

Field name	Type	Rqd/Opt	Description
`fulfillment-info`	Array of volume-fulfillment-info objects	Required	The information required to fulfill one or more FICON volumes. Each element in the array is a volume-fulfillment-info nested object defined in the next table.

Each nested volume-fulfillment-info object contains the following fields:

Field name	Type	Description
`storage-volume-uri`	String/URI	The canonical URI of the storage volume that is being fulfilled.
`control-unit-uri`	String/URI	The canonical URI of the storage control unit in which the backing ECKD volume is defined.
`unit-address`	String (2)	A two-character lower case hexadecimal number that represents the unit address of the backing ECKD volume within the storage control unit.

Description

This operation provides information about the ECKD storage resources that have been selected by a SAN administrator to fulfill one or more base storage volumes that are part of a request for new or modified FICON storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to those storage resources.

Once a FICON storage volume has been fulfilled, the storage control unit and unit that has been assigned to it cannot be changed. If the **size**, **cylinders** or **model** properties of a volume are modified, the **fulfillment-state** of the volume is changed to **"pending"**, indicating that action may be necessary by the storage administrator to adjust the size of the backing ECKD volume. The **fulfillment-state** of the volume is changed back to **"complete"** by fulfilling the volume again with the exact same **control-unit-uri** and **unit-address**, indicating that the changes have been completed.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or any **storage-volume-uri** in a nested object within the **fulfillment-info** request field does not identify a storage volume in the storage group, or any **control-unit-uri** in a nested object within the **fulfillment-info** request field does not identify a storage control unit, or the current value of the group's **type** property is not **"fc"**.

If the storage volume's **fulfillment-state** property value is not **"pending"**, or if the CPC on which this storage group resource exists is not active, or if the storage group's **connectivity** property value does not equal the number of storage paths configured in any identified storage control unit, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if any identified unit address is not defined on the identified control unit, or if any backing ECKD volume is an alias volume or has already being used to fulfill another storage volume, or if any identified control unit and unit address would remap a previously fulfilled storage volume.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each of the storage volumes is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. The **control-unit-uri** and **unit-address** properties for each storage volume are set to the same-named values in the nested object that references that volume. The **size** and **model** property values for each storage volume are copied to the **active-size** and **active-model** properties. Change notifications for each storage volume's **fulfillment-state**, **control-unit-uri**, **unit-address**, **active-size**, and **active-model** properties are emitted asynchronously to this operation. Change notification for the parent storage group's **fulfillment-state** property may be also emitted.

If an unexpected error occurs when processing a valid request, a 500 (Server Error) status code is returned. In such a case, the targeted storage volumes may be left in a mixed state, with some being fulfilled and others not. The **error-details** field in the response body will contain an array of storage volume URIs that identify the storage volumes that were not fulfilled.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 260. Fulfill FICON Storage Volumes: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – Storage Administrator** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	2	The storage control unit identified by **control-unit-uri** in any volume-fulfillment-info nested object does not exist on the HMC.
	4	The storage group object with the **object-id** {storage-group-id} has a **type** value other than **"fc"**.
	445	A storage volume identified by **storage-volume-uri** in any volume-fulfillment-info nested object does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	471	The current value of the **fulfillment-state** property of the storage volume identified by **storage-volume-uri** in any volume-fulfillment-info nested object is not **"pending"**.
	473	A storage volume identified by **storage-volume-uri** in any volume-fulfillment-info nested object has been previously fulfilled, and the values of the associated **control-unit-uri** and **unit-address** nested fields do not exactly match the previous values.
	484	A unit address identified by **unit-address** in any volume-fulfillment-info nested object is not a member of a volume group on the control unit identified by the associated **control-unit-uri**.
	485	The unit address identified by **unit-address** in any volume-fulfillment-info nested object in the control unit identified by the associated **control-unit-uri** has already been used to fulfill another FICON storage volume.
	486	The unit address identified by **unit-address** in any volume-fulfillment-info nested object is a member of an alias volume group on the control unit identified by the associated **control-unit-uri**.
	492	The **connectivity** property value of a storage volume identified by **storage-volume-uri** in any volume-fulfillment-info nested object does not equal the number of storage paths configured in the storage control unit identified by the associated **control-unit-uri**.

Table 260. Fulfill FICON Storage Volumes: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
500 (Server Error)	Various	An unexpected error occurred when processing a valid request. The **error-details** field in the response body will contain a **failed-storage-volume-uris** field with an array of canonical URIs of the storage volumes that were not fulfilled.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/operations/fulfill-ficon-storage-
volumes HTTP/1.1
x-api-session: 1m1og3c15tzef37kr5z9q5gugo6cqk959sfyzd4tjr03oy4vfw
content-type: application/json
content-length: 514
{
   "fulfillment-info":[
      {
         "control-unit-uri":"/api/storage-control-units/4423c72a-057e-11ea-82b4-fa163e2a9969",
         "storage-volume-uri":"/api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/
           storage-volumes/b336cb04-057d-11ea-8f2d-fa163e2a9969",
         "unit-address":"00"
      },
      {
         "control-unit-uri":"/api/storage-control-units/44a4b8b2-057e-11ea-82b4-fa163e2a9969",
         "storage-volume-uri":"/api/storage-groups/b2e50a3a-057d-11ea-8f2d-fa163e2a9969/
           storage-volumes/b3381982-057d-11ea-8f2d-fa163e2a9969",
         "unit-address":"01"
      }
   ]
}
```

Figure 293. Fulfill FICON Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 12 Nov 2019 19:08:28 GMT

<No response body>
```

Figure 294. Fulfill FICON Storage Volumes: Response

Usage notes

There is a limit on the request body size for the Fulfill FICON Storage Volumes operation of 256KB. This puts an effective limit of about 1050 volumes that can be fulfilled with a single request.

Fulfill FCP Storage Volume

The operation`Fulfill FCP Storage Volume` provides information about the storage resource that has been selected by a SAN administrator to fulfill a boot storage volume that is part of a request for new FCP storage.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/storage-volumes/{storage-volume-id}/operations/
   fulfill-fcp-storage-volume
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object and the URI variable *{storage-volume-id}* is the element ID of the storage volume element.

Request body contents

Field name	Type	Rqd/Opt	Description
`world-wide-port-name`	String (16)	Required	A 16-character lower case hexadecimal string that contains the world wide port name of the FCP storage controller containing the SCSI boot device.
`logical-unit-number`	String (16)	Required	A 16-character lower case hexadecimal string that contains the logical unit number (LUN) of the SCSI boot device.
`adapter-port-uri`	String/ URI	Required	The canonical URI of the storage adapter port that is to be assigned to the storage volume.

Description

This operation provides information about the storage resource that has been selected by a SAN administrator to fulfill a storage volume that is part of a request for new FCP storage. Until this information is provided, the CPC does not have all the data it needs to address and connect to that storage resource. Only FCP boot volumes may be manually fulfilled. Other volumes will be automatically fulfilled when the CPC detects communication paths to them. FCP volumes may be manually fulfilled only before the CPC discovers a working communications path, indicated by a **"validated" status** on at least one of the parent storage group's WWPNs. After that point, all volumes in the group are automatically fulfilled and manual fulfillment is no longer possible.

If the API user does not have action/task permission to the **Configure Storage – Storage Administrator** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or the element ID *{storage-volume-id}* does not identify a storage volume in the group, or the **adapter-port-uri** field does not identify an adapter object to which the API user has object-access permission or identifies an adapter with a **type** value other than **"fcp"**, or the current value of the group's **type** property is not **"fcp"**, or the current value of the volume's **usage** property is not **"boot"**.

If the **status** property of at least one WWPN in the parent group's **world-wide-port-names** property value is **"validated"**, or if the **adapter-port-uri** field references a storage adapter port that does not reside in the target storage group's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the storage volume's **fulfillment-state** property is changed to **"complete"**. Change notification for the storage volume's **fulfillment-state** property is emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to the adapter containing the storage adapter port identified by the **adapter-port-uri** field.
- Action/task permission to the **Configure Storage – Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 261. Fulfill FCP Storage Volume: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – Storage Administrator** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	2	The adapter object containing the storage adapter port referenced in the **adapter-port-uri** field does not exist on the HMC, or the API user does not have object-access permission to it.
	4	The storage group object with the **object-id** {storage-group-id} has a **type** value other than **"fcp"**.
	5	A storage volume with **element-id** *{storage-volume-id}* does not exist in the storage group on the HMC.
	6	The adapter containing the storage adapter port referenced in the **adapter-port-uri** field has a **type** value other than **"fcp"**.
	442	The storage volume object with the **object-id** *{storage-group-id}* has a usage value other than **"boot"**.
409 (Conflict)	2	The parent storage group with **object-id** *{storage-group-id}* was busy and the request timed out.
	441	The storage adapter port referenced by the **adapter-port-uri** field resides in a different CPC than the targeted storage group.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/
  2d71eeae-9fe9-11e8-b163-fa163e3c2af4/operations/fulfill-fcp-storage-volume HTTP/1.1
x-api-session: 2315cbdr840lgk09zmkpsfzwykutdinjxz1u1sxqxa0chulv2v
content-type: application/json
content-length: 177
{
   "adapter-port-uri":"/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
   "logical-unit-number":"93e4118a8b42c859",
   "world-wide-port-name":"a1b2c3d4e5f60002"
}
```

Figure 295. Fulfill FCP Storage Volume: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:49:28 GMT

<No response body>
```

Figure 296. Fulfill FCP Storage Volume: Response

Usage notes

It is not required to manually fulfill FCP storage volumes. If the `Fulfill FCP Storage Volume` request is not made targeting an FCP boot volume, the WWPN and LUN will be assigned algorithmically when the volume is sensed. Manual fulfillment can optionally be performed to speed up the process when the WWPN and LUN information is known, for example when the LUN configuration is done automatically.

Accept Mismatched Storage Volumes

The `Accept Mismatched Storage Volumes` operation completes the fulfillment process for selected volumes of a storage group that have been flagged as possibly being mismatched or overprovisioned.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/accept-mismatched-storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`storage-volume-uris`	Array of String/URI	Required	The canonical URIs of the mismatched storage volumes that are to be accepted.

Description

This operation accepts FCP or NVMe storage volumes that have been flagged as being mismatched or overprovisioned.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object

ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or one of the storage volumes identified in the **storage-volume-uris** field in the request body does not identify a storage volume in the storage group, or the current value of the group's **type** property is not **"fcp"** or **"nvme"**.

If the target storage group is busy, or if one of the storage volumes identified in the **storage-volume-uris** field has a **fulfillment-state** property value that is not **"pending-with-mismatches"** or **"overprovisioned"**, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each storage volume identified in the **storage-volume-uris** field is changed to **"complete"**. If the **fulfillment-state** property values for all other storage volumes in the parent storage group are also **"complete"**, the parent storage group's **fulfillment-state** property will also be set to **"complete"**. Change notifications for each storage volume's **fulfillment-state** property are emitted asynchronously to this operation. A change notification for the parent storage group's **fulfillment-state** property may be also emitted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 262. Accept Mismatched Storage Volumes: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the **object-id** *{storage-group-id}* has a **type** value other than **"fcp"** or **"nvme"**.
	5	A storage volume identified in the **storage-volume-uris** array field does not exist in the storage group with the **object-id** *{storage-group-id}*.
409 (Conflict)	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	471	The current value of the **fulfillment-state** property of one of the storage volumes identified in the **storage-volume-uris** array field is not **"pending-with-mismatches"** or **"overprovisioned"**.

Table 262. Accept Mismatched Storage Volumes: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/operations/accept-mismatched-
   storage-volumes HTTP/1.1
x-api-session: 5b4tzc4fkakbrg4qsprt1ha84vvwdauch9y2x7al5wew0ann58
content-type: application/json
content-length: 138
{
   "storage-volume-uris":[
      "/api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/storage-volumes/cb61f800-
         e29e-11e8-9879-fa163e9c462b"
   ]
}
```

Figure 297. Accept Mismatched Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 07 Nov 2018 15:42:06 GMT

<No response body>
```

Figure 298. Accept Mismatched Storage Volumes: Response

Reject Mismatched FCP Storage Volumes

The `Reject Mismatched FCP Storage Volumes` operation rejects selected volumes of an FCP storage group that have been flagged as being possibly mismatched or overprovisioned.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/reject-mismatched-storage-volumes
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`storage-volume-uris`	Array of String/URI	Required	The canonical URIs of the mismatched storage volumes that are to be rejected.

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified through email of the rejected storage group resources. These email addresses will appear in the "to:" address list in the email that is sent. If not specified, no email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the rejected storage group resources. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be omitted when the **email-to-addresses** field is omitted.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the rejected storage group resources. The text can include HTML formatting tags. The **email-insert** field must be omitted when the **email-to-addresses** field is omitted. If not specified, an email without a special text insert will be sent.

Description

This operation rejects FCP storage volumes that have been flagged as being mismatched or overprovisioned. The storage administrator is requested to change the storage configuration to delete the identified volumes.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or one of the storage volumes identified in the **storage-volume-uris** field in the request body does not identify a storage volume in the storage group, or the current value of the group's **type** property is not **"fcp"**.

If the target storage group is busy, or if one of the storage volumes identified in the **storage-volume-uris** field has a **fulfillment-state** property value that is not **"pending-with-mismatches"** or **"overprovisioned"**, or if an email address is specified when the target console does not support sending emails, a 409 (Conflict) status code is returned.

If the request body fails to validate, or if the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the **fulfillment-state** property of each storage volume identified in the **storage-volume-uris** field is changed to **"deleting"**. Change notifications for each storage volume's **fulfillment-state** property are emitted asynchronously to this operation.

If the **email-to-addresses** field is present in the request body, an email containing information about the storage volume resources that have been rejected is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the changes to the volume fulfillment states. An

API client should assume that the **fulfillment-state** property of each storage volume was changed to **"deleting"** even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 263. Reject Mismatched FCP Storage Volumes: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group object with the **object-id** *{storage-group-id}* has a **type** value other than **"fcp"**.
	5	A storage volume identified in the **storage-volume-uris** array field does not exist in the storage group with the **object-id** *{storage-group-id}*.
409 (Conflict)	2	The storage group object with the **object-id** *{storage-group-id}* was busy and the request timed out.
	471	The current value of the **fulfillment-state** property of one of the storage volumes identified in the **storage-volume-uris** array field is not **"pending-with-mismatches"** or **"overprovisioned"**.
	491	An error occurred when sending the email. This failure applies only to the ending of the email. If this reason code is returned, the fulfillment state of all storage volumes will have been modified.
	493	The **email-to-addesses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/operations/reject-mismatched-
   storage-volumes HTTP/1.1
x-api-session: 3q85scprm0jy2cxmt86ordrc5typ19zpiy5szvxeb0d4duwzz8
content-type: application/json
content-length: 138
{
   "storage-volume-uris":[
      "/api/storage-groups/bc600cca-e29e-11e8-9879-fa163e9c462b/storage-volumes/fe7829bc-
         e29e-11e8-8ee9-fa163e9c462b"
   ]
}
```

Figure 299. Reject Mismatched FCP Storage Volumes: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 07 Nov 2018 15:44:38 GMT

<No response body>
```

Figure 300. Reject Mismatched FCP Storage Volumes: Response

List Virtual Storage Resources of a Storage Group

The `List Virtual Storage Resources of a Storage Group` operation lists the virtual storage resources of the FCP storage group with the given identifier.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/virtual-storage-resources
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`device-number`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **device-number** property.
`adapter-port-uri`	String/URI	Optional	Filter string to limit returned objects to those that have a matching **adapter-port-uri** property. Specify an empty query parameter value to select objects that have an **adapter-port-uri** property value of null, for example: "adapter-port-uri="
`partition-uri`	String/URI	Optional	Filter string to limit returned objects to those that have a matching **partition-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`virtual-storage-resources`	Array of virtual-storage-resource-info objects	Array of virtual-storage-resource-info objects, described in the next table. The returned array may be empty.

Each nested virtual-storage-resource-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The canonical URI path (**element-uri**) of the virtual Storage Resource element object.
`name`	String	The **name** property of the virtual storage resource element.
`device-number`	String	The **device-number** property of the virtual storage resource element.
`adapter-port-uri`	String/ URI	The **adapter-port-uri** property of the virtual storage resource element.
`partition-uri`	String/ URI	The **partition-uri** property of the virtual storage resource element.

Description

This operation lists the virtual storage resources that are owned by the identified storage group. The element URI, name, device number and associated adapter port and partition are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or does not identity an FCP storage group, a 404 (Not Found) status code is returned

If the **name** or **device-number** query parameter is specified, the returned list is limited to those virtual storage resources that have a **name** or **device-number** property matching the specified filter pattern. If the **name** or **device-number** parameter is omitted, the filtering on the omitted property name is not done.

If the **adapter-port-uri** or **partition-uri** query parameter is specified, the returned list is limited to those virtual storage resources that have a matching **adapter-port-uri** or **partition-uri** property. If the **adapter-port-uri** or **partition-uri** parameter is omitted, the filtering on the omitted property name is not done.

If no virtual storage resources are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 594.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 264. List Virtual Storage Resources of a Storage Group: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage group with the object ID *{storage-group-id}* does not exist on the HMC, or is a storage group of the wrong **type**, or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources
  HTTP/1.1
x-api-session: 46zfjkryyht0wrf7v8fjxfmza830u42yuob8p0ct727mr2so2x
```

Figure 301. List Virtual Storage Resources of a Storage Group: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:50:54 GMT
content-type: application/json;charset=UTF-8
content-length: 737
{
   "virtual-storage-resources":[
      {
         "adapter-port-uri":"/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/
           storage-ports/0",
         "device-number":"0000",
         "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
           virtual-storage-resources/4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
         "name":"vhba_FCP Storage Group0",
         "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4"
      },
      {
         "adapter-port-uri":"/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/
           storage-ports/0",
         "device-number":"0001",
         "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
           virtual-storage-resources/4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
         "name":"vhba_FCP Storage Group1",
         "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4"
      }
   ]
}
```

Figure 302. List Virtual Storage Resources of a Storage Group: Response

Get Virtual Storage Resource Properties

The `Get Virtual Storage Resource Properties` operation retrieves the properties of a single Virtual Storage Resource element object.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group object and the URI variable *{virtual-storage-resource-id}* is the element ID of the virtual storage resource element.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Virtual Storage Resource object as defined in the "Data model" on page 525. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the virtual storage resource object as defined in "Virtual Storage Resource element object" on page 542.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, or does not identity an FCP storage group, or if the element ID *{virtual-storage-resource-id}* does not identify a virtual storage resource in the storage group.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 597.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 265. Get Virtual Storage Resource Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A storage group with **object-id** *{storage-group-id}* does not exist on the HMC, or is a storage group of the wrong **type**, or the API user does not have object-access permission to it.
	5	A virtual storage resource with **element-id** *{virtual-storage-resource-id}* does not exist in the storage group on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
  4ce7feea-9fe9-11e8-b6db-fa163e3c2af4 HTTP/1.1
x-api-session: 63naxg5mxrpmugoyatmcwn6nv5zz5lbltb1zf6ga1408yhplxi
```

Figure 303. Get Virtual Storage Resource Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:51:56 GMT
content-type: application/json;charset=UTF-8
content-length: 690
{
   "adapter-port-uri":"/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
   "class":"virtual-storage-resource",
   "degraded-reasons": [
     "adapter"
   ],
   "description":"",
   "device-number":"0001",
   "element-id":"4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
   "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
     virtual-storage-resources/4ce7feea-9fe9-11e8-b6db-fa163e3c2af4",
   "name":"vhba_FCP Storage Group1",
   "parent":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
   "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4",
   "world-wide-port-name":"a1b2c3d4e5f60003",
   "world-wide-port-name-info":{
     "status":"validated",
     "world-wide-port-name":"a1b2c3d4e5f60003"
   }
}
```

Figure 304. Get Virtual Storage Resource Properties: Response

Update Virtual Storage Resource Properties

The `Update Virtual Storage Resource Properties` operation updates one or more of the writable properties of a virtual storage resource.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/virtual-storage-resources/{virtual-storage-resource-
id}
```

In this request, the URI variable *{storage-group-id}* is the object ID of the storage group object and the URI variable *{virtual-storage-resource-id}* is the element ID of the virtual storage resource element.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Virtual Storage Resource element object" on page 542. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a virtual storage resource's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission, or does not identity an FCP storage group, or if the element ID *{virtual-storage-resource-id}* does not identify a Virtual Storage Resource element object in the storage group. In addition to object-access to the parent storage group, updates to the **adapter-port-uri** or **device-number** properties also require object-access permission to the partition that is associated with the target virtual storage resource. If these properties appear in the request body and the user does not have object-access to the partition identified in the **partition-uri** property, a 403 (Forbidden) status code is returned. If the CPC on which this virtual storage resource exists is not active, or if the **adapter-port-uri** field references a storage adapter port that does not reside in the target virtual storage resource's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines an **adapter-port-uri** field that references a non-FCP adapter, because the parent storage group already contains a virtual storage resource with the specified name, or because the partition associated with this virtual storage resource already contains a device with the supplied device number.

If the request body contents are valid, the virtual storage resource's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Object-access permission to the partition referenced by the virtual storage resource's **partition-uri** property. This requirement only applies when updating the **adapter-port-uri** or **device-number** properties.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 266. Update Virtual Storage Resource Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	8	A virtual storage resource with the **name** specified in the request body already exists within the parent storage group.
	18	The request body contains the **adapter-port-uri** field, but the **type** property of the parent adapter is not **"fcp"**.
	452	The new value supplied in the device-number field conflicts with an existing device number for another device attached to the partition associated with this virtual storage resource.

Table 266. Update Virtual Storage Resource Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
	450	The **adapter-port-uri** or **device-number** fields are present in the request body and the API user does not have object-access permission to the partition referenced by the current value of the virtual storage resource's **partition-uri** property
404 (Not Found)	1	A storage group with **object-id** *{storage-group-id}* does not exist on the HMC, or is a storage group of the wrong **type**, or the API user does not have object-access permission to it.
	5	A virtual storage resource with **element-id** *{virtual-storage-resource-id}* does not exist in the storage group on the HMC.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The parent storage group with **object-id** *{storage-group-id}*or the partition identified by this virtual storage resource's **partition-uri** property value was busy and the request timed out.
	6	The **status** property value for the partition identified by this virtual storage resource's **partition-uri** property is not valid to perform the operation (must be in one of the following states: **"active"**, **"degraded"**, **"paused"**, **"reservation-error"**, **"stopped"**, or **"terminated"**).
	441	The storage adapter port referenced by the **adapter-port-uri** field resides in a different CPC than the targeted virtual storage resource.
	498	The **adapter-port-uri** field was present in the request body when the current value of that property in the virtual storage resource element with **object-id** *{virtual-storage-resource-id}* is **null**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
  4ce7feea-9fe9-11e8-b6db-fa163e3c2af4 HTTP/1.1
x-api-session: 3f4ojo8469608kk498sr9e96l05yrsyapyheb1icwn355ws0pd
content-type: application/json
content-length: 44
{
   "description":"vHBA in FCP Storage Group"
}
```

Figure 305. Update Virtual Storage Resource Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 14 Aug 2018 18:49:28 GMT

<No response body>
```

Figure 306. Update Virtual Storage Resource Properties: Response

Get Partitions for a Storage Group

The `Get Partitions for a Storage Group` operation lists the partitions to which the storage group with the given identifier is attached.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/operations/get-partitions
```

In this request, the URI variable *{storage-group-id}* is the **object-id** of the Storage Group object.

Query parameters

Name	Type	Req/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned Partition objects to those that have a matching **name** property
`status`	String Enum	Optional	Optional filter string to limit returned Partition objects to those that have a matching **status** property. Value must be a valid partition **status** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`partitions`	Array of partition-info objects	Array of partition-info objects, described in the next table. The returned array may be empty.

Each nested partition-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The canonical URI path (**object-uri**) of the partition object. This property will be null when the current user has no object access permission to the partition.
`name`	String	The **name** property of the object.
`status`	String Enum	The **status** property of the object.

Description

This operation lists the partitions to which the identified storage group is attached. The object URI, name and status are provided for each.

If the object ID *{storage-group-id}* does not identify a storage group object to which the API user has object-access permission, a 404 (Not Found) status code is returned

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid value for the **status** property according to the partition data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have the specified **status** value. If the **status** parameter is omitted, this filtering is not done.

The operation lists all partitions attached to the storage group including partitions the current user may not have object access permission to. For partitions without object access permission, the **object-uri** property will be null. If no partitions are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 601.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 267. Get Partitions for a Storage Group: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	A storage group with **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-groups/ec638c1e-9689-11e8-aa30-fa163e27d492/operations/get-partitions
 HTTP/1.1
x-api-session: 40xd7zji2d7manim658hz4ot8r7x7fp3xuk0nyuced9psqubc8
```

Figure 307. Get Partitions for a Storage Group: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Aug 2018 15:53:46 GMT
content-type: application/json;charset=UTF-8
content-length: 235
{
   "partitions":[
      {
         "name":"Partition 2",
         "object-uri":"/api/partitions/8d1ca9a0-998d-11e8-8352-fa163e27d492",
         "status":"active"
      },
      {
         "name":"Partition 1",
         "object-uri":"/api/partitions/72a0f8b0-998d-11e8-8352-fa163e27d492",
         "status":"stopped"
      }
   ]
}
```

Figure 308. Get Partitions for a Storage Group: Response

Validate LUN Path

The `Validate LUN Path` operation verifies if a LUN is reachable along the path specified. The path is specified by the host worldwide port name, adapter port, target worldwide port name, and the LUN to indicate the connections from the CPC to the Storage Subsystem.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/validate-lun-path
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC that contains the host worldwide port name and the adapters.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`host-world-wide-port-name`	String (16)	Required	A 16-character lower-case hexadecimal string that contains a WWPN that uniquely identifies a host or initiator device in the SAN.
`adapter-port-uri`	String/ URI	Required	The canonical URI path of the Storage Port element object to be used to validate the LUN configuration.
`target-world-wide-port-name`	String (16)	Required	A 16-character lower-case hexadecimal string that contains a WWPN that uniquely identifies a target port in one of the storage subsystems connected to this CPC.
`logical-unit-number`	String (1-16)	Required	The lower-case hexadecimal logical unit number (LUN) representing the storage device configured in the storage controller.

Description

This operation verifies if a LUN identified by **logical-unit-number** configured for the **host-world-wide-port-name** through the storage port designated by **adapter-port-uri** and the target port designated by **target-world-wide-port-name.**

The **adapter-port-uri** field in the request body must designate an existing Storage Port of an existing adapter, and the API user must have object-access permission to that adapter. If not, 404 (Not Found) status code is returned

If the **adapter-port-uri** does not designate an existing Storage Port of an existing adapter of **type "fcp"**, a 409 (Conflict) status code is returned.

\If the storage controller does not configure the specified **target-world-wide-port-name** for the given combination of **host-world-wide-port-name** and **adapter-port-uri**, a 409 (Conflict) status is returned.

If the storage controller does not configure the specified **logical-unit-number** for the given combination of **host-world-wide-port-name**, **adapter-port-uri** and **target-world-wide-port-name**, a 409 (Conflict) status code is returned.

If the CPC does not have **"dpm-storage-management"** feature enabled, or designates a Storage Port that does not reside in the target CPC, a 409 (Conflict) status is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Object-access permission to the adapter containing the port designated by the **adapter-port-uri** field.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 268. Validate LUN Path: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
404 (Not Found)	2	The storage port designated by the **adapter-port-uri** does not exist on the HMC or the API user does not have object-access permission for the adapter containing the storage port.

Table 268. Validate LUN Path: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	13	The operation is not supported when the **"dpm-storage-management"** feature is not enabled on the CPC.
	441	The storage adapter port referenced by the **adapter-port-uri** field resides in a different CPC than the targeted storage group.
	483	The adapter designated by the **adapter-port-uri** is not of **type "fcp"**.
	484	A configuration for the **target-world-wide-port-name** does not exist in the storage controller for the given **host-world-wide-port-name** and **adapter-port-uri** combination.
	485	A configuration for the **logical-unit-number** does not exist in the storage controller for the given **host-world-wide-port-name**, **adapter-port-uri** and **target-world-wide-port-name** combination.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e/operations/validate-lun-path
Request Headers   : {
content-type:application/json,
x-api-session:p5e86fr0btfsspy95sal06vkfp4cbkj4tjq27qmlr3fcx71dd
content-length:233,
{
       "adapter-port-uri":"/api/adapters/df4d482e-ce2c-11e7-bca6-00106f0d81cb/
          storage-ports/0",
    "host-world-wide-port-name":"C05076FFE8000002",
    "logical-unit-number":"4013402F00000000",
    "target-world-wide-port-name":"C05076FFE8001001"
}
```

Figure 309. Validate LUN Path: Request

```
204 No Content
cache-control: no-cache,
date:Mon, 30 Jul 2018 17:32:40 GMT,
server:Hardware management console API web server / 2.0,
x-request-id:Sxb597d98c-8bef-11e8-8db3-00106f0d8409.1fca Rxc,
x-wsa-provider-duration-ms:934

<No response body>
```

Figure 310. Validate LUN Path: Response

Start FCP Storage Discovery

The `Start FCP Storage Discovery` operation triggers asynchronous storage discovery for an FCP storage group. The system will check the connections for the storage group and create a connection report indicating the status of those connections. Because the system periodically performs storage discovery for FCP storage groups and creates a connection report for those storage groups, it is not

necessary to use this operation; however, issuing this operation may cause the discovery to be performed sooner than it would have otherwise.

HTTP method and URI

```
POST /api/storage-groups/{storage-group-id}/operations/start-fcp-storage-discovery
```

In this request, the URI variable *{storage-group-id}* is the object ID of the target storage group.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`force-restart`	Boolean	Optional	Indicates if there is an in-progress discovery operation for the specified storage group, it should be terminated and started again. If **false** or there is no in-progress discovery operation for the specified storage group, a new one is started. Default: **false**

Response body contents

Once the start request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 607. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The operation triggers Logical Unit Number (LUN) discovery for the target storage group. Once the LUN discovery operation is completed a connection report is generated for the storage group, indicating the status of the connections. Once the connection report is created, the `Get Connection Report` operation can be used to retrieve it. This operation is only valid for the storage groups of **type "fcp"**. The API user must have action/task permission to the **Configure Storage - System Programmer** task or the **Configure Storage - Storage Administrator** task; otherwise, status code 403 (Forbidden) is returned.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission or if the storage group identified by the object ID *{storage-group-id}* is not of **type "fcp"**.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or the **Configure Storage - Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 606.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 269. Start FCP Storage Discovery: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage - Storage Administrator** tasks.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group with object ID *{storage-group-id}* gas a **type** value other than **"fcp"**.
409 (Conflict)	1	The state of the CPC to which the storage group is associated is not valid to perform the operations. It must be one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 270. Start FCP Storage Discovery: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	The operation completed successfully
404 (Not Found)	1	The storage group with **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
500 (Server Error)	263	The storage discovery operation failed and there will be no automatic retry.

Example HTTP interaction

```
POST /api/storage-groups/2217a632-56bb-11e9-a434-fa163effe469/operations/
   start-fcp-storage-discovery HTTP/1.1
x-api-session: 36lkftjan75oyr72g1bz9bic4r68qk2dtsm0svw2ei9tpa2az4
```

Figure 311. Start FCP Storage Discovery: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location:/api/jobs/31500aa4-56bb-11e9-ac0a-fa163efc2ef4
cache-control : no-cache
date: Thu, 04 Apr 2019 09:22:38 GMT
content-type:application/json;charset=UTF-8
content-length:60
{
   "job-uri":"/api/jobs/31500aa4-56bb-11e9-ac0a-fa163efc2ef4"
}
```

Figure 312. Start FCP Storage Discovery: Response

Usage notes

- At any point in time, there will be at most one in-progress discovery operation checking connections for a given storage group. While a discovery operation is in progress, any new request for the same storage group from the APIs or the Graphical User Interface will be linked to that in-progress discovery operation. An in-progress discovery operation can be interrupted for various reasons, such as starting a partition to which this storage group is attached, reassigning adapters of the virtual resources of the storage group, or invoking the `Start FCP Storage Discovery` operation with a **force-restart** flag set to **true**. If an in-progress discovery operation is interrupted for any reason, the operation will be automatically retried and the previous requests associated with that in-progress discovery operation will be linked to the new operation.
- If the request fails for any reason, all requests associated with that in-progress request will fail and an error code will be returned in the asynchronous result.
- If the HMC stops communicating with the Support Element or if the CPC enters into an invalid state, when the discovery operation for a storage group is in progress, the operation will be retried when the HMC starts communicating with the SE again or if the CPC enters a valid state, respectively. The valid CPC states for the discovery operation are **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.

Get Connection Report

The `Get Connection Report` operation retrieves the most recent connection report for a storage group. The connection report for a storage group of **type "fc"** is first generated when at least one of the volumes is mapped and the storage group is attached to at least one partition. The connection report for a storage group of **type "fcp"** is first generated when a storage discovery operation completes successfully for the first time. For a storage group of **type "fcp"**, the worldwide port names referenced in the connection report are the initiator worldwide port names assigned to the virtual storage resources of the storage group.

HTTP method and URI

```
GET /api/storage-groups/{storage-group-id}/operations/get-connection-report
```

In this request, the URI variable *{storage-group-id}* is the object ID of the Storage Group object.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`last-scan-time`	Timestamp	The time the last discovery operation for a storage group completed successfully. This field is only present when the storage group's **type** property value is **"fcp"**.
`storage-group-uri`	String/ URI	The canonical URI path of the storage group with which the connection report is associated.
`fcp-fabrics`	Array of fcp-fabric-info objects	The list of information about the distinct storage fabrics that the storage adapters in the system are connected. Each element in the array is an instance of an fcp-fabric-info object, defined in Table 271 on page 609. This field is only present when the storage group's **type** property is **"fcp"**.
`fcp-storage-subsystems`	Array of fcp-storage-subsystem-info objects	The list of information about the distinct storage subsystems, where the FCP volumes for this storage group are defined. Each element in the array is an instance of fcp-storage-subsystem-info object, defined in Table 274 on page 611. This field is only present when the storage group's **type** property is **"fcp"**.
`ficon-storage-subsystems`	Array of ficon-storage-subsystem-info objects	The list of information about the distinct storage subsystems, where the FICON volumes for this storage group are defined. Each element in the array is an instance of a ficon-storage-subsystem-info object, defined in Table 276 on page 612. This field is only present when the storage group's **type** property is **"fc"**.

An fcp-fabric-info object contains information about a single fabric configured in the storage area network switches. Each fabric-info object contains the following fields:

Table 271. fcp-fabric-info nested object

Name	Type	Description
fabric-id	String (16)	The World Wide Name (WWN) of the uplink Fibre Channel Switch.
expected-adapter-count	Integer	The number of adapters expected to be zoned for the worldwide port names (WWPNs) of the storage group in this fabric.
accessible-subsystem-ids	Array of String	The list of worldwide node names (WWNNs) of the storage subsystems that are accessible for the adapters in this fabric. Each WWNN is a 16-character lowercase hexadecimal string representing the node name defined in the storage subsystem.
world-wide-port-names	Array of world-wide-port-name-zone-info objects	The list of information about the worldwide port names (WWPNs) that have been allocated to support this FCP storage group. Each element in the array is an instance of world-wide-port-name-zone-info object, defined in Table 272 on page 610.

Table 271. fcp-fabric-info nested object (continued)		
Name	**Type**	**Description**
unzoned-adapters	Array of adapter-info objects	The list of information about storage adapters in the system that are connected to this fabric but does not contain a zoning configuration or any of the WWPNs in this storage group. Each element in the array is an instance of adapter-info object, defined in Table 273 on page 610.

A world-wide-port-name-zone-info object contains information about the World Wide Port Names (WWPNs) of the storage group and the set of adapters that are zoned for it in the fabric which can be used to connect to the storage network. Each world-wide-port-name-zone-info object contains the following fields:

Table 272. world-wide-port-name-zone-info nested object		
Name	**Type**	**Description**
world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains the world wide port name used to access the FCP storage volumes defined for the storage group.
zoned-adapters	Array of adapter-info objects	The list of information about storage adapters that are configured in one of the active zones of the storage fabric. Each element in the array is an instance of the adapter-info object, defined in Table 273 on page 610. The world wide port name can be assigned to the virtual storage resource created on the adapter to access the FCP storage volumes defined for the storage group. The sum of the number of distinct adapters zoned for this world wide port name across various fabrics should be greater than or equal to the value of the **connectivity** field of the storage group.

An adapter-info object contains information about an adapter. Each adapter-info object contains the following fields:

Table 273. adapter-info nested object		
Name	**Type**	**Description**
adapter-name	String (1-64)	The display name specified for this adapter
adapter-status	String Enum	The status of the adapter. The adapter can be used to check the connections only when the **status** is **"active"**.
adapter-uri	String/ URI	The canonical URI path of the Adapter object.

An fcp-storage-subsystem-info object contains information about a storage subsystem that the FCP volumes of the storage group are defined in. Each fcp-storage-subsystem-info object contains the following fields:

Table 274. fcp-storage-subsystem-info nested object		
Name	**Type**	**Description**
world-wide-node-names	Array of string	The list of worldwide node names (WWNNs) defined for the storage subsystem. Each WWNN is a 16-character lowercase hexadecimal string representing the node name defined in the storage subsystem.
world-wide-port-names-count	Integer	The number of worldwide port names (WWPNs) that are zoned and masked correctly in the storage switch and storage subsystem, to access the expected number of FCP volumes for the storage group.
expected-volumes-count	Integer	The number of FCP volumes expected to be discovered in this storage subsystem.
storage-configurations	Array of storage-configuration-info objects	The list of information about subsystem configurations for each worldwide port name of the storage group. Each element in the array is an instance of storage-configuration-info object, defined in Table 275 on page 611.

A storage-configuration-info object contains information about the subsystem configuration for each WWPN of the storage group. Each storage-configuration-info object contains the following fields:

Table 275. storage-configuration-info nested object		
Name	**Type**	**Description**
world-wide-port-name	String (16)	A 16-character lowercase hexadecimal string that contains the worldwide port name used to access the FCP storage volumes defined for the storage group.
discovered-volumes-count	Integer	The number of volumes the WWPN is configured to access in the storage subsystem.
volumes-configuration-status	String Enum	The configuration status of the WWPN across the subsystems. Indicates if the WWPN is able to access the expected number of volumes in the storage subsystem. Possible values: • **"no-volumes"** - The WWPN cannot access any volume across the subsystems. • **"too-many-volumes"** - The WWPN can access more volumes than what was expected for the storage group. • **"too-few-volumes"** - The WWPN can access fewer volumes than what was expected for the storage group. • **"different-volumes"** - The WWPN can access one or more volumes not defined to the storage group, or it cannot access one or more of the volumes that are defined to the storage group, but not all. • **"correct-volumes"** - The WWPN can access all of the defined volumes of the storage group and no other volumes.

A ficon-storage-subsystem-info object contains information about a storage subsystem to which the FICON volumes of the storage group are mapped. Each ficon-storage-subsystem-info object contains the following fields:

Table 276. ficon-storage-subsystem-info nested object		
Name	**Type**	**Description**
mapped-subsystem-uri	String/ URI	The canonical URI path of the Storage Subsystem object that contains control units mapped to the volumes of the storage group.
mapped-control-unit-uris	Array of String/ URI	The list of storage control units within the storage subsystem that are mapped to the volumes in the storage group. Each element in this array is the canonical URI path of a Storage Control Unit object.

Description

This operation retrieves the most recent connection report of the specified storage group. The API user must have action/task permission to the **Configure Storage – System Programmer** task or the **Configure Storage – Storage Administrator** task; otherwise, status code 403 (Forbidden) is returned.

A 404 (Not Found) status code is returned if the object ID *{storage-group-id}* does not identify a Storage Group object to which the API user has object-access permission.

A 409 (Conflict) status code is returned if a connection report has not been generated for the storage group and the user can request one using the `Start FCP Storage Discovery` operation for a storage group of **type "fcp"**.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage group whose **object-id** is *{storage-group-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 609.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 277. Get Connection Report: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage - Storage Administrator** tasks.
404 (Not Found)	1	A storage group with the **object-id** *{storage-group-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	The storage group with object ID *{storage-group-id}* gas a **type** value other than **"fcp"**.
409 (Conflict)	250	A connection report has not been generated.

Table 277. Get Connection Report: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/operations/
  get-connection-report HTTP/1.1
x-api-session: 1wianh609dwilk4lfgyriv8ywx4o2okf4dl9kn3jln1p9d1msw
```

Figure 313. Get Connection Report: Request

```
200 OK
server: Hardware management console API web server/2.0
cache-control: no-cache
date: Thu ,04 Apr 2019 19:36:08 GMT
content-type:application/json;charset=UTF-8
content-length:3623
{
 "fcp-fabrics":[
  {
   "accessible-subsystem-ids":[
    [
     "5005076400c00003",
     "5005076400c00002",
     "5005076400c00004",
     "5005076400c00001",
     "5005076400c00000"
    ]
   ],
   "expected-adapter-count":1,
   "fabric-id":"100099906",
   "unzoned-adapters":[],
   "world-wide-port-names":[
    {
     "world-wide-port-name":"a1b2c3d4e5f60045",
     "zoned-adapters":[
      {
       "adapter-name":"FCP 0131 Z01B-17",
       "adapter-status":"active",
       "adapter-uri":"/api/adapters/4b3c435a-448d-11e9-9536-fa163effe469"
      }
     ]
    }
   ]
  },
  {
   "accessible-subsystem-ids":[
    [
     "5005076400c00003",
     "5005076400c00002",
     "5005076400c00004",
     "5005076400c00001",
     "5005076400c00000"
    ]
   ],
```

Figure 314. Get Connection Report: Response (Part 1)

```
    "expected-adapter-count":0,
    "fabric-id":"100099901",
    "unzoned-adapters":[],
    "world-wide-port-names":[
     {
      "world-wide-port-name":"a1b2c3d4e5f60045",
      "zoned-adapters":[
       {
        "adapter-name":"FCP 0101 Z01B-02",
        "adapter-status":"active",
        "adapter-uri":"/api/adapters/45b2e5a6-448d-11e9-9536-fa163effe469"
       }
      ]
     }
    ]
   },
   {
    "accessible-subsystem-ids":[
     [
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
     ]
    ],
    "expected-adapter-count":0,
    "fabric-id":"100099905",
    "unzoned-adapters":[],
    "world-wide-port-names":[
     {
      "world-wide-port-name":"a1b2c3d4e5f60045",
      "zoned-adapters":[
       {
        "adapter-name":"FCP 0130 Z01B-17",
        "adapter-status":"active",
        "adapter-uri":"/api/adapters/4aaaf260-448d-11e9-9536-fa163effe469"
       }
      ]
     }
    ]
   },
   {
    "accessible-subsystem-ids":[
     [
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
     ]
    ],
```


Figure 315. Get Connection Report: Response (Part 2)

```
    "expected-adapter-count":0,
    "fabric-id":"100099904",
    "unzoned-adapters":[],
    "world-wide-port-names":[
     {
      "world-wide-port-name":"a1b2c3d4e5f60045",
      "zoned-adapters":[
       {
        "adapter-name":"FCP 0105 Z01B-03",
        "adapter-status":"active",
        "adapter-uri":"/api/adapters/a5a6bb22-448d-11e9-ad2d-fa163effe469"
       }
      ]
     }
    ]
   },
   {
    "accessible-subsystem-ids":[
     [
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
     ]
    ],
    "expected-adapter-count":0,
    "fabric-id":"100099903",
    "unzoned-adapters":[],
    "world-wide-port-names":[
     {
      "world-wide-port-name":"a1b2c3d4e5f60045",
      "zoned-adapters":[
       {
        "adapter-name":"FCP 0100 Z01B-02",
        "adapter-status":"active",
        "adapter-uri":"/api/adapters/3ee50ee8-448d-11e9-9536-fa163effe469"
       }
      ]
     }
    ]
   },
   {
    "accessible-subsystem-ids":[
     [
      "5005076400c00003",
      "5005076400c00002",
      "5005076400c00004",
      "5005076400c00001",
      "5005076400c00000"
     ]
    ],
```


Figure 316. Get Connection Report: Response (Part 3)

```
    "expected-adapter-count":0,
    "fabric-id":"100099902",
    "unzoned-adapters":[],
    "world-wide-port-names":[
     {
      "world-wide-port-name":"a1b2c3d4e5f60045",
      "zoned-adapters":[
       {
        "adapter-name":"FCP 0104 Z01B-03",
        "adapter-status":"active",
        "adapter-uri":"/api/adapters/a317e23c-448d-11e9-ad2d-fa163effe469"
       }
      ]
     }
    ]
   }
  ],
  "fcp-storage-subsystems":[
   {
    "expected-volumes-count":1,
    "storage-configurations":[
     {
      "discovered-volumes-count":5,
      "volumes-configuration-status":"too-many-volumes",
      "world-wide-port-name":"a1b2c3d4e5f60045"
     }
    ],
    "world-wide-node-names":[
     "5005076400c00003",
     "5005076400c00002",
     "5005076400c00004",
     "5005076400c00001",
     "5005076400c00000"
    ],
    "world-wide-port-names-count":1
   }
  ],
  "last-scan-time":1554370171000,
  "storage-group-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469"
 }
```


Figure 317. Get Connection Report: Response (Part 4)

Usage notes

- If a connection report has not been generated for a storage group of **type "fcp"**, use the `Start FCP Storage Discovery` operation to initiate the creation of the connection report.
- If a connection report has not been generated for a storage of **type "fc"**, use the `Fulfill Storage Volume` operation to map the ECKD storage resources to the volumes that are part of the storage group and the `Attach Storage Group to Partition` operation to attach the storage group to a partition.
- For a storage group of **type "fcp"**, the following conditions should be satisfied for a valid connection to the storage resources:
  - The number of common adapter objects in the **zoned-adapters** field for all **world-wide-port-names** in each of the fabrics defined in the **fcp-fabric-info** object should be greater than or equal to the **expected-adapter-count** value in the **fcp-fabric-info** field .
  - For a given **world-wide-port-name**, the sum of the **discovered-volumes-count** value across the **storage-configuration-objects** should be greater than or equal to the number of volumes that are part of the storage request.
  - The **volumes-configuration-state** in each **storage-configuration-info** object indicates the status of configuration for a worldwide port name across subsystems. For an ideal connection the **volumes-configuration-state** should either be **"correct-volumes"** or **"too-many-volumes"** for all the worldwide port names.

Get Storage Group Histories

The `Get Storage Group Histories` operation returns a chronological list of actions performed on the storage groups known to the target Console. The response can be filtered according to query parameters, if specified.

HTTP method and URI

```
GET /api/console/operations/get-storage-group-histories
```

Query parameters:

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp to filter the entries returned in the **storage-group-actions** list. Actions with **action-time** values earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp to filter the entries returned in the **storage-group-actions** list. Actions with **action-time** values later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`cpc-uri`	String/URI	Optional	The canonical URI path of a CPC object to limit the returned entries to those that have a matching **cpc-uri** field. If not specified, then no such filtering is performed.
`storage-group-uri`	String/URI	Optional	The canonical URI path of a Storage Group object to limit the returned entries to those that have a matching **storage-group-uri** field. If not specified, then no such filtering is performed.
`user-name`	String (4-320)	Optional	Filter pattern (regular expression) to limit the entries returned in the **storage-group-actions** list to those that have a matching **user-name** field. If not specified, then no such filtering is performed.
`storage-group-name`	String	Optional	Filter pattern (regular expression) to limit the returned entries to those that have a matching **storage-group-name** field. If not specified, then no such filtering is performed.
`storage-group-type`	String	Optional	Filter string to limit returned entries to those that have a matching **storage-group-type** field. If not specified, then no such filtering is performed. Value must be a valid **storage-group-type** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`storage-group-histories`	Array of storage-group-history-info objects	The list of information about the history of actions performed on each storage group. Each element in the array is an instance of storage-group-history-info object, described in the next table. The returned array may be empty.

Each storage-group-history-info object contains the following fields:

Table 278. storage-group-history-info nested object

Name	Type	Description of specialization
cpc-uri	String/URI	The canonical URI path of the CPC object with which the storage group is associated.
storage-group-uri	String/URI	The canonical URI path of the Storage Group object with which the storage group history is associated.
storage-group-actions	Array of storage-group-action-info objects	The list of information about the actions performed on the storage group. Each element in the array is an instance of a storage-group-action-info object, described in the next table.
storage-group-name	String (1-64)	The display name specified for the storage group with which the storage group history is associated. Refer to the **name** property of the Storage Group object defined in Table 235 on page 526.
storage-group-type	String Enum	The type of storage resources managed by the storage group with which the storage group history is associated. Refer to the **type** property of the Storage Group object in Table 236 on page 526

Each storage-group-action-info object contains the following fields:

Table 279. storage-group-action-info nested object

Name	Type	Description
user-name	String (4-320)	The name of the user who performed the action on the storage group. Refer to the **name** property of the User object, defined in Table 456 on page 893 The value is **"no user"** if the action was performed by the system.
action-time	Timestamp	The time when the action was performed on the storage group.

Table 279. storage-group-action-info nested object (continued)		
Name	**Type**	**Description**
action	String Enum	The action that was performed on the storage group. Values: • **"creation-requested"** - Storage group was created. • **"modification-requested"** - One or more properties of the storage group were modified. Some properties are changed immediately and some are pending fulfillment which is indicated by the field **storage-group-fulfillment-state**. • **"deletion-requested"** - Storage group was deleted. • **"request-fulfilled"** - Storage group request was fulfilled. • **"attached-to-partition"** - Storage group was attached to a partition. • **"detached-from-partition"** - Storage group was detached from a partition. • **"auto-created"** - Storage group was created automatically when the CPC was upgraded to a level where **"dpm-storage-management"** feature is enabled. • **"auto-created-during-migration"** - Storage group was created as part of MES upgrade. • **"mismatched-volumes-found"** - Volumes not matching the storage request for a storage group of **type "fcp"** were discovered. • **"mismatched-volumes-accepted"** - Mismatched volumes discovered were accepted by the user. • **"mismatched-volumes-rejected"** - Mismatched volumes discovered were rejected by the user. • **"send-email-failed"** - Send email operation failed during creation/modification/deletion of the storage group. • **"manually-fulfilled"** - A storage volume was manually fulfilled. • **"incomplete"** - Fulfillment state of storage group changed to Incomplete. • **"incomplete-resolved"** - Fulfillment state of storage group changed from Incomplete. • **"backing-adapter-changed"** – Backing adapter of a virtual storage resource belonging to the storage group was changed. • **"partition-name-changed"** – The name of a Partition to which this storage group is attached was changed. • **"device-number-changed"** - Device number of virtual storage resource or a storage volume of **type "fc"** or **"nvme"** belonging to the storage group was changed. • **"ficon-volumes-mapped"** - Storage volumes of **type "fc"** were mapped to control units. • **"nvme-ssd-removed"** - A solid-state drive (SSD) is removed from an adapter that is associated with a volume in a storage group of **type "nvme"**.

Table 279. storage-group-action-info nested object (continued)		
Name	**Type**	**Description**
		• **"nvme-ssd-installed"** - A solid-state drive (SSD) of same size is installed in to an adapter that is associated with a volume in a storage group of **type "nvme"**. • **"nvme-volume-mismatch"** - A solid-state drive (SSD) is plugged in to an adapter that is associated with a volume in a storage group of **type "nvme"**, but does not match the volume's size. • **"nvme-volumes-accepted"** - One or more volumes in a storage group of **type "nvme"** were updated to match the properties of the SSD that is currently plugged in to the adapter.
storage-group-fulfillment-state	String Enum	The fulfillment state of the storage group, after the action was performed. Values: • Any value of the **fulfillment-state** property as defined in the Storage Group Data model in Table 236 on page 526. • **"deleted"** - The storage group has been deleted.
storage-group-configuration	storage-group-configuration object	A nested object that provides additional information about the configuration of the storage group that was changed as part of this action. The value is a single instance of a storage-group-configuration object, defined in Table 280 on page 621. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the value of the **action** property is one of the following: **"creation-requested"**, **"modification-requested"**, **"deletion-requested"**, **"backing-adapter-changed"**, **"mismatched-volumes-found"**, **"mismatched-volumes-rejected"**, or **"auto-created-during-migration"**. When the action is **"creation-requested"**, the fields in this object will be populated similar to the **"modification-requested"** action, considering all values of the create request as new or added, When the action is **"deletion-requested"**, the fields in this object will be populated similar to the **"modification-requested"** action , considering all values of the delete request as old or deleted.
partition-uri	String/URI	The canonical URI path of the partition which this storage group is attached to or detached from. This field is only present if the **action** property is either **"attached-to-partition"**, **"detached-from-partition"**, or **"partition-name-changed"**.
old-partition-name	String (1-64)	The old name of the partition to which the storage group is attached. This field is only present if the **action** property is **"partition-name-changed"**.

Table 279. storage-group-action-info nested object (continued)		
Name	**Type**	**Description**
new-partition-name	String (1-64)	The new name of the partition to which the storage group is attached. This field is only present if the **action** property is **"partition-name-changed"**.

Each storage-group-configuration object contains the following fields:

Table 280. storage-group-configuration nested object		
Name	**Type**	**Description**
old-name	String (1-64)	The old value for the **name** property of the storage group.
new-name	String (1-64)	The new value for the **name** property of the storage group.
old-description	String (0-200)	The old value for the **description** property of the storage group.
new-description	String (0-200)	The new value for the **description** property of the storage group.
old-shared	Boolean	The old value for the **shared** property of the storage group.
new-shared	Boolean	The new value for the **shared** property of the storage group.
old-connectivity	Integer	The old value for the **connectivity** property of the storage group. This field is only present when the storage group's **type** property is **"fcp"** or **"fc"**.
new-connectivity	Integer	The new value for the **connectivity** property of the storage group. This field is only present when the storage group's **type** property is **"fcp"** or **"fc"**.
old-max-partitions	Integer	The old value for the **max-partitions** property of the storage group.
new-max-partitions	Integer	The new value for the **max-partitions** property of the storage group.
old-direct-connection-count	Integer (0-1000)	The old value for the**direct-connection-count** property of the storage group. This field is only present when the storage group's **type** property is **"fcp"**.
new-direct-connection-count	Integer (0-1000)	The new value for the **direct-connection-count** property of the storage group. This field is only present when the storage group's **type** property is **"fcp"**.
world-wide-port-names-added	Array of world-wide-port-name-info objects	The list of information about the worldwide port names (WWPNs) that have been newly allocated to this storage group. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 240 on page 544. This field is only present when the action resulted in new worldwide port names being added for this storage group.

Table 280. storage-group-configuration nested object (continued)		
Name	**Type**	**Description**
world-wide-port-names-deleted	Array of world-wide-port-name-info objects.	The list of information about the worldwide port names (WWPNs) that have been deleted from this storage group. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 240 on page 544. This field is only present when the action resulted in worldwide port names being deleted from this storage group.
storage-volumes-added	Array of storage-volume-info objects	The list of information about the storage volumes that were newly added to this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 281 on page 623. This field is only present when the action resulted in new volumes being added to the storage group. When the **action** is **"mismatched-volumes-found"**, this field contains the volumes that were sensed but did not match the storage request.
storage-volumes-modified	Array of storage-volume-info objects	The list of information about the storage volumes that were modified in this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 281 on page 623. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the action resulted in volumes of this storage group being modified. When the **action** is **"mismatched-volumes-accepted"**, this field contains the volumes that were accepted by the `Accept Mismatched Storage Volumes` operation.
storage-volumes-deleted	Array of storage-volume-info objects	The list of information about the storage volumes that were deleted from this storage group. Each element in this array is an instance of a storage-volume-info object, defined in Table 281 on page 623. This field is only present when the action resulted in volumes being deleted from this storage group. When the **action** is **"mismatched-volumes rejected"**, this field contains the volumes that were rejected by the `Reject Mismatched FCP Storage Volumes` operation.
virtual-storage-resources-added	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were added. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 282 on page 625. This field is only present when the storage group's **type** property is **"fcp"** and the **action** property is either **"attached-to-partition"** or **"modification-requested"** and action resulted in virtual storage resources of this storage group being added.

Table 280. storage-group-configuration nested object (continued)		
Name	**Type**	**Description**
virtual-storage-resources-modified	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were modified. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 282 on page 625. The fields in this object are present only when the storage group's **type** property is **"fcp"** and the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the **action** property is either **"device-number-changed"**, **"backing-adapter-changed"**, or **"modification-requested"** and action resulted in virtual storage resources of this storage group being modified.
virtual-storage-resources-deleted	Array of virtual-storage-resource-info objects	The list of information about the virtual storage resources of this storage group that were deleted. Each element in this array is an instance of a virtual-storage-resource-info object, defined in Table 282 on page 625. This field is only present when the storage group's **type** property is **"fcp"** and the **action** property is either **"detached-from-partition"** or **"modification-requested"** and action resulted in virtual storage resources of this storage group being deleted.
storage-volumes-in-error	Array of storage-volume-info objects	The list of information about the storage volumes that were in error state. Each element in this array is an instance of a storage-volume-info object, defined in Table 281 on page 623. This field is only present when the storage group's type property is **"nvme"** and the action is either **"nvme-ssd-plugged-out"** or **"nvme-volume-mismatch"**, and when the action resulted in some of the volumes going into an error state.
storage-volumes-recovered	Array of storage-volume-info objects	The list of information about the storage volumes that were recovered from error state. Each element in this array is an instance of a storage-volume-info object, defined in Table 281 on page 623. This field is only present when the storage group's type property is **"nvme"** and the action is either **"nvme-ssd-plugged-in"** or **"nvme-volumes-accepted"**, and the action resulted in some of the volumes recovered from an error state.

Each storage-volume-info object contains the following fields:

Table 281. storage-volume-info nested object		
Name	**Type**	**Description**
element-uri	String/URI	The canonical URI path for the storage volume element object that was modified in this action.
old-name	String (1-64)	The old value for the **name** property of the storage volume.
new-name	String (1-64)	The new value for the **name** property of the storage volume.
old-description	String (0-100)	The old value for the **description** property of the storage volume.

Table 281. storage-volume-info nested object (continued)		
Name	**Type**	**Description**
new-description	String (0-100)	The new value for the **description** property of the storage volume.
fulfillment-state	String Enum	The value of the **fulfillment-state** property of the storage volume at the time of this action. This field is always present.
eckd-type	String Enum	The value of the **eckd-type** property of the storage volume at the time of this action. This field is always present when the storage group's **type** property is **"fc"**.
sensed-uuid	String (16, 32)	The value of the **sensed-uuid** property of the storage volume at the time of this action. This field is only present when the storage group's **type** property is **"fcp"** and the volume has already been sensed.
old-size	Float	The old value for the **size** property of the storage volume.
new-size	Float	The new value for the **size** property of the storage volume.
old-usage	String Enum	The old value for the **usage** property of the storage volume.
new-usage	String Enum	The new value for the **usage** property of the storage volume.
old-device-number	String (4)	The old value for the **device-number** property of the storage volume. This field is only present when the **storage-group-type** property is **"fc"** or **"nvme"**. The value will be **null** when the device number was not previously specified for the storage volume.
new-device-number	String (4)	The new value for the **device-number** property of the storage volume. This field is only present when the **storage-group-type** property is **"fc"** or **"nvme"**. The value will be **null** when the device number was not specified for the storage volume.
adapter-uri	String/URI	The value of the **adapter-uri** property of the storage volume. This field is only present when the storage group's **type** property is **"nvme"**.
old-serial-number	String	The old value for the **serial-number** property of the storage volume. This field is only present when the storage group's **type** property is **"nvme"**.
new-serial-number	String	The new value for the **serial-number** property of the storage volume. This field is only present when the storage group's **type** property is **"nvme"**.

Table 281. storage-volume-info nested object (continued)		
Name	**Type**	**Description**
control-unit-uri	String/URI	The value of the **control-unit-uri** property of the storage volume at the time of this action. This field is always present when the storage group's **type** property is **"fc"**. The value will be **null** if a **"base"** volume was not fulfilled at the time of this action.
unit-address	String (2)	The value of the **unit-address** property of the storage volume at the time of this action. This field is always present when the storage group's **type** property is **"fc"**. The value will be **null** if a **"base"** volume was not fulfilled at the time of this action.

Each virtual-storage-resource-info object contains the following fields:

Table 282. virtual-storage-resource-info nested object		
Name	**Type**	**Description**
element-uri	String/URI	The canonical URI path for the Virtual Storage Resource element object that was modified in this action.
old-name	String (1-64)	The old value for the **name** property of the virtual storage resource.
new-name	String (1-64)	The new value for the **name** property of the virtual storage resource.
old-description	String (0-1024)	The old value for the **description** property of the virtual storage resource.
new-description	String (0-1024)	The new value for the **description** property of the virtual storage resource.
partition-uri	String/URI	The value of the **partition-uri** property of the Virtual Storage Resource element object at the time of this action. This field is always present.
old-device-number	String (4)	The old value for the **device-number** property of the virtual storage resource. The value will be **null** when the device number was not previously specified for the virtual storage resource.
new-device-number	String (4)	The new value for the **device-number** property of the virtual storage resource. The value will be **null** when the device number was not specified for the virtual storage resource.
old-adapter-port-uri	String/URI	The old value for the **adapter-port-uri** property of the virtual storage resource. The value will be **null** when the **adapter-port-uri** was not previously specified for the virtual storage resource.

Table 282. virtual-storage-resource-info nested object (continued)		
Name	**Type**	**Description**
new-adapter-port-uri	String/URI	The new value for the **adapter-port-uri** property of the virtual storage resource. The value will be **null** when the **adapter-port-uri** was not specified for the virtual storage resource.
world-wide-port-name	String (16)	The value of the **world-wide-port-name** property of the Virtual Storage Resource element object at the time of this action.

Description

This operation returns information about the history of actions performed for all the storage groups known to this console. It contains a chronological list of actions performed on storage groups. Information about storage groups to which the API user does not have object-access permission, are not included in the response. The history entry for each storage group contains an array of storage-group-action-info objects representing the actions performed on the storage group.

The order in which the storage-group-history-info objects are returned is not guaranteed and can change for every request.

A storage group's history entries are not deleted when the storage group object is deleted. The history entries are preserved for 30 days from the deletion of the storage group object, after which they are deleted automatically. This allows API clients to query minimal configuration information for the storage group even after they receive notification that the storage group has been deleted. API clients can use this configuration information to undo any configuration done earlier on the SAN in order to fulfill this storage group.

The entries can be limited by specifying explicit filtering criteria on the request. Filtering can occur on two different levels. The **storage-group-uri** and **cpc-uri** query parameters limit the results to a single storage group, or to all storage groups associated with a specific CPC respectively. The **begin-time**, **end-time** and **user-name query** parameters filter the actions that are returned for each storage group.

If the **begin-time** query parameter is specified, then any actions earlier than that time are omitted. If the **end-time** query parameter is specified, then any actions later than that time are omitted.

If the **storage-group-uri** query parameter is specified for the request, the returned list is limited to the storage group history that has a matching **storage-group-uri** value.

If the **cpc-uri** query parameter is specified for the request, the returned list is limited to the storage group history entries that have a matching **cpc-uri** value.

If the **user-name** query parameter is specified for the request, only actions with a **user-name** property matching the specified filter pattern are returned.

If the **storage-group-name** parameter is specified for the request, the returned list is limited to the storage group history entries that have a **storage-group-name** property matching the specified filter pattern.

If the **storage-group-type** parameter is specified for the request, the returned list is limited to the storage group history entries that have a **storage-group-type** property matching the specified value.

If no storage group history entries are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Storage Group object designated by the **storage-group-uri** property of each storage-group-history-info object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 617.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 283. Get Storage Group Histories: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter contains an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-storage-group-histories?storage-group-
  name=SG_CFI_con_22 HTTP/1.1
x-api-session: 2x828y1ic1qtyenkg33luihglj9qlnyxdilblc2q14y5bus44s
```

Figure 318. Get Storage Group Histories: Request

```
200 OK
server: Hardware management console API web server/2.0
cache-control: no-cache
date: Thu, 04 Apr 2019 09:41:54 GMT
content-type:application/json;charset=UTF-8
content-length: 5473
{
 "storage-group-histories":[
  {
   "cpc-uri":"/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
   "storage-group-actions":[
    {
     "action":"creation-requested",
     "action-time":1554370624684,
     "storage-group-configuration":{
      "new-connectivity":1,
      "new-description":"",
      "new-direct-connection-count":0,
      "new-max-partitions":1,
      "new-name":"SG_CFI_con_22",
      "new-shared":false,
      "storage-volumes-added":[
       {
        "element-uri":"/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469/
          storage-volumes/30bc6360-56bd-11e9-a59c-fa163effe469",
        "fulfillment-state":"pending",
        "new-description":"",
        "new-name":"vol1",
        "new-size":10.0,
        "new-usage":"boot",
        "sensed-uuid":null
       }
      ],
      "world-wide-port-names-added":[
       {
        "status":"not-validated",
        "world-wide-port-name":"a1b2c3d4e5f60046"
       }
      ]
     },
     "storage-group-fulfillment-state":"pending",
     "user-name":"sysprog"
    },
```

Figure 319. Get Storage Group Histories: Response (Part 1)

```
    {
     "action":"deletion-requested",
     "action-time":1554370628053,
     "storage-group-configuration":{
      "old-connectivity":1,
      "old-description":"",
      "old-direct-connection-count":0,
      "old-max-partitions":1,
      "old-name":"SG_CFI_con_22",
      "old-shared":false,
      "storage-volumes-deleted":[
       {
        "element-uri":"/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469/storage-volumes/
          30bc6360-56bd-11e9-a59c-fa163effe469",
        "fulfillment-state":"pending",
        "old-description":"",
        "old-name":"vol1",
        "old-size":10.0,
        "old-usage":"boot",
        "sensed-uuid":null
       }
      ],
      "world-wide-port-names-deleted":[
       {
        "status":"not-validated",
        "world-wide-port-name":"a1b2c3d4e5f60046"
       }
      ]
     },
     "storage-group-fulfillment-state":"deleted",
     "user-name":"sysprog"
    }
   ],
   "storage-group-name":"SG_CFI_con_22",
   "storage-group-type":"fcp",
   "storage-group-uri":"/api/storage-groups/3063183c-56bd-11e9-a59c-fa163effe469"
  },
  {
   "cpc-uri":"/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
   "storage-group-actions":[
    {
     "action":"creation-requested",
     "action-time":1554370175411,
     "storage-group-configuration":{
      "new-connectivity":1,
      "new-description":"",
      "new-direct-connection-count":0,
      "new-max-partitions":1,
      "new-name":"SG_CFI_con_22",
      "new-shared":false,
      "storage-volumes-added":[
```


Figure 320. Get Storage Group Histories: Response (Part 2)

```
        {
         "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
           24f274b2-56bc-11e9-bcc3-fa163effe469",
         "fulfillment-state":"pending",
         "new-description":"",
         "new-name":"vol1",
         "new-size":10.0,
         "new-usage":"boot",
         "sensed-uuid":null
        }
       ],
       "world-wide-port-names-added":[
        {
         "status":"not-validated",
         "world-wide-port-name":"a1b2c3d4e5f60045"
        }
       ]
      },
      "storage-group-fulfillment-state":"pending",
      "user-name":"sysprog"
     },
     {
      "action":"mismatched-volumes-found",
      "action-time":1554370171053,
      "storage-group-fulfillment-state":"pending-with-mismatches",
      "user-name":"no user"
     },
     {
      "action":"deletion-requested",
      "action-time":1554370194985,
      "storage-group-configuration":{
       "old-connectivity":1,
       "old-description":"",
       "old-direct-connection-count":0,
       "old-max-partitions":1,
       "old-name":"SG_CFI_con_22",
       "old-shared":false,
       "storage-volumes-deleted":[
        {
         "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
           26225f50-56bc-11e9-8d03-fa163effe469",
         "fulfillment-state":"overprovisioned",
         "old-description":"",
         "old-name":"Data 48",
         "old-size":0.0,
```


Figure 321. Get Storage Group Histories: Response (Part 3)

```
        "old-usage":"data",
        "sensed-uuid":"6005076307FFC6A60000000000000001"
       },
       {
        "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
          26182224-56bc-11e9-8d03-fa163effe469",
        "fulfillment-state":"overprovisioned",
        "old-description":"",
        "old-name":"Data 47",
        "old-size":0.0,
        "old-usage":"data",
        "sensed-uuid":"6005076307FFC6A60000000000000002"
       },
       {
        "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
          24f274b2-56bc-11e9-bcc3-fa163effe469",
        "fulfillment-state":"pending-with-mismatches",
        "old-description":"",
        "old-name":"vol1",
        "old-size":10.0,
        "old-usage":"boot",
        "sensed-uuid":"6005076307FFC6A60000000000000004"
       },
       {
        "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
          262c85c0-56bc-11e9-8d03-fa163effe469",
        "fulfillment-state":"overprovisioned",
        "old-description":"",
        "old-name":"Data 49",
        "old-size":0.0,
        "old-usage":"data",
        "sensed-uuid":"6005076307FFC6A60000000000000000"
       },
       {
        "element-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469/storage-volumes/
          260cd37e-56bc-11e9-8d03-fa163effe469",
        "fulfillment-state":"overprovisioned",
        "old-description":"",
        "old-name":"Data",
        "old-size":0.0,
```


Figure 322. Get Storage Group Histories: Response (Part 4)

```
        "old-usage":"data",
        "sensed-uuid":"6005076307FFC6A60000000000000003"
       }
      ],
      "world-wide-port-names-deleted":[
       {
        "status":"validated",
        "world-wide-port-name":"a1b2c3d4e5f60045"
       }
      ]
     },
     "storage-group-fulfillment-state":"deleted",
     "user-name":"sysprog"
    }
   ],
   "storage-group-name":"SG_CFI_con_22",
   "storage-group-type":"fcp",
   "storage-group-uri":"/api/storage-groups/249cbe64-56bc-11e9-bcc3-fa163effe469"
  },
  {
   "cpc-uri":"/api/cpcs/f4556a31-dcf7-344e-a770-0e68783cac23",
   "storage-group-actions":[
    {
     "action":"creation-requested",
     "action-time":1554370914499,
     "storage-group-configuration":{
      "new-connectivity":1,
      "new-description":"",
      "new-direct-connection-count":0,
      "new-max-partitions":1,
      "new-name":"SG_CFI_con_22",
      "new-shared":false,
      "storage-volumes-added":[
       {
        "element-uri":"/api/storage-groups/dd29557c-56bd-11e9-9081-fa163effe469/storage-volumes/
          dd7851ea-56bd-11e9-9081-fa163effe469",
        "fulfillment-state":"pending",
        "new-description":"",
        "new-name":"vol1",
        "new-size":10.0,
        "new-usage":"boot",
        "sensed-uuid":null
       }
      ],
      "world-wide-port-names-added":[
       {
        "status":"not-validated",
        "world-wide-port-name":"a1b2c3d4e5f60047"
       }
      ]
     },
     "storage-group-fulfillment-state":"pending",
     "user-name":"sysprog"
    }
   ],
   "storage-group-name":"SG_CFI_con_22",
   "storage-group-type":"fcp",
   "storage-group-uri":"/api/storage-groups/dd29557c-56bd-11e9-9081-fa163effe469"
  }
 ]
}
```


Figure 323. Get Storage Group Histories: Response (Part 5)

Inventory service data

Information about the Storage Groups managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for storage group objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class "storage-group" are to be included. Information for a particular storage group is included only if the API user has object-access permission to that object.

For each storage group to be included, the inventory response array includes the following:

- An array entry for the storage group object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for "Get Storage Group Properties" on page 556. That is, the data provided is the same as would be provided if a `Get Storage Group Properties` operation were requested targeting this object.
- An array entry for each storage volume element associated with the storage group. For each such storage volume, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for "Get Storage Volume Properties" on page 578.
- An array entry for each virtual storage resource element associated with the storage group. For each such virtual storage resource, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for "Get Virtual Storage Resource Properties" on page 597.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage group. These objects would appear as multiple array entries in the response array:

```
{
   "active-connectivity":null,
   "active-max-partitions":null,
   "candidate-adapter-port-uris":[
      "/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
      "/api/adapters/f4f1479c-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0"
   ],
   "class":"storage-group",
   "connectivity":2,
   "cpc-uri":"/api/cpcs/3aadfe48-128c-3766-8f51-37738f784591",
   "description":"",
   "direct-connection-count":0,
   "fulfillment-state":"pending",
   "max-partitions":2,
   "name":"FCP Group",
   "object-id":"2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
   "object-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
   "parent":"/api/console",
   "shared":true,
   "storage-volume-uris":[
      "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/
        2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
      "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/storage-volumes/
        2d71eeae-9fe9-11e8-b163-fa163e3c2af4"
   ],
   "type":"fcp",
   "unassigned-world-wide-port-names":[
      {
         "status":"not-validated",
         "world-wide-port-name":"a1b2c3d4e5f60006"
      },
      {
         "status":"not-validated",
         "world-wide-port-name":"a1b2c3d4e5f60005"
      }
   ],
   "virtual-storage-resource-uris":[
      "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
        4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
      "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-resources/
        4ce7feea-9fe9-11e8-b6db-fa163e3c2af4"
   ]
},
{
   "active-size":null,
   "class":"storage-volume",
   "description":"",
   "element-id":"2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
   "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/
     storage-volumes/2d6355ba-9fe9-11e8-b163-fa163e3c2af4",
   "fulfillment-state":"pending",
   "name":"1.00 GiB Boot",
   "parent-storage-group-uris":[
      "/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4"
   ],
   "size":1.0,
   "usage":"boot",
   "uuid":"a1b2c3d4e5f6000325d4677ab2320e54"
},
```


Figure 324. Storage Group object: Sample inventory data - Response (Part 1)

```
{
   "adapter-port-uri":"/api/adapters/f0f668e8-9fe7-11e8-bc9a-fa163e3c2af4/storage-ports/0",
   "class":"virtual-storage-resource",
   "description":"",
   "device-number":"0000",
   "element-id":"4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
   "element-uri":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4/virtual-storage-
     resources/4ca56526-9fe9-11e8-b6db-fa163e3c2af4",
   "name":"vhba_FCP Group0",
   "parent":"/api/storage-groups/2cbc7c2c-9fe9-11e8-b163-fa163e3c2af4",
   "partition-uri":"/api/partitions/4bf93d46-9fe9-11e8-b6db-fa163e3c2af4",
   "world-wide-port-name":"a1b2c3d4e5f60002"
   "world-wide-port-name-info":{
     "status":"not-validated",
     "world-wide-port-name":"a1b2c3d4e5f60002"
     }
}
```


Figure 325. Storage Group object: Sample inventory data - Response (Part 2)

Storage Template object

A Storage Template object represents a single storage template associated with a DPM-enabled CPC. A storage template can be used when creating a storage group to initialize the group's property values to the current values of the template's properties. See `Create Storage Group`. The Storage Template object APIs provide access to the set of storage templates that are associated with a CPC that is enabled for DPM. APIs exist to create and delete storage templates, query storage templates, and modify selected properties of storage templates. APIs also exist to list and query properties of the storage template volume elements of storage templates.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 285 on page 636 . The Storage Template object does not support the operational status related properties.

Table 284. Storage Template object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Storage Template object is of the form `/api/storage-templates/{storage-template-id}` where `{storage-template-id}` is the value of the **object-id** property of the Storage Template object.
object-id	—	String (36)	The unique identifier for the storage template instance.
parent	—	String/ URI	The parent of a storage template is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (16)	The class of a Storage Template object is **"storage-template"**.
name	(w)(pc)	String (1-64)	The display name specified for the storage template. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage templates associated to the same CPC.

Table 284. Storage Template object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the storage template. This value will be used to initialize the description property of storage groups that are created using this storage template. Default value: An empty string.

Class specific additional properties

In addition to the properties defined through included schema, this object includes the following additional class-specific properties:

Table 285. Storage Template object: class specific properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this Storage Template object. This value will be used to initialize the **cpc-uri** property of the storage groups that are created using this storage template.	All
type	—	String Enum	The type of storage resources managed by the storage template. Values: • **"fcp"** - Fibre Channel Protocol • **"fc"** - Fibre Connection This value will be used to initialize the **type** property of storage groups that are created using this storage template.	All
shared	(w)(pc)	Boolean	**true** if storage groups created from this template can be attached to more than one partition; **false** if the storage group is dedicated to a single partition. This property's value must be **"false"** when the value of the **max-partitions** property is **1**. This value will be used to initialize the shared property of storage groups that are created using this storage template. Default value: **true**.	All

Table 285. Storage Template object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
storage-volume-uris	(c)(pc)	Array of String/ URI	The list of storage template volumes for this storage template. Each element in this array is the canonical URI path of a Storage Template Volume object. This value will change when storage template volumes are added or removed from this storage template through the `Modify Storage Template Properties` operation. This value will be used to initialize the **storage-volume-uris** property of storage groups that are created using this storage template. Default value: An empty array.	All
connectivity	(w)(pc)	Integer (1-255)	The number of adapters to utilize for storage groups created from this template. When the **type** value is **"fc"**, the maximum value for the **connectivity** property is **8**. This value will be used to initialize the **connectivity** property of storage groups that are created using this storage template. Default value: **2** for storage templates of **type "fcp"**; **8** for storage templates of **type "fc"**.	All
max-partitions	(w)(pc)	Integer	The maximum number of partitions to which this storage template can be attached. The value of **max-partitions** cannot exceed **1** if the value of the **shared** field is **false**. The value of **max-partitions** must be greater than **1** if the value of the **shared** field is **true**. This value will be used to initialize the **max-partitions** property of storage groups that are created using this storage template. The minimum value for the **max-partitions** property is **1**; the maximum is the value of the CPC object's **maximum-partitions** property. Default value: **2**.	fcp

Table 285. Storage Template object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
direct-connection-count	(w)(pc)	Integer (0-1000)	The number of additional virtual storage resource connections for the host that can be directly assigned to a guest virtual machine for storage groups created from this template. A value of zero indicates this feature is disabled. The **direct-connection-count** property cannot be enabled (greater than zero) when the **shared** property is **true**. This value will be used to initialize the **direct-connection-count** property of storage groups that are created using this storage template. Default value: **0**.	fcp

Storage Template Volume element object

A Storage Template Volume element object defines the size and usage of a single storage template volume within its parent storage template.

Table 286. Storage Template Volume element object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
element-id	—	String (36)	Unique identifier for the storage template volume instance	All
element-uri	—	String/ URI	The canonical URI path for the storage template volume element object, of the form `/api/storage-templates/{storagetemplate-id}/storage-template-volumes/{storage-template-volume-id}`, where *{storage-template-id}* is the object-id of the containing storage template, and *{storage-template-volume-id}* is the **element-id** of this storage template volume.	All
parent	—	String/ URI	The parent of a storage template volume is its owning storage group, so the parent value is the canonical URI path for the storage template.	All
class	—	String (23)	The class of a Storage Template Volume element object is **"storage-template-volume"**.	All

Table 286. Storage Template Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
name	(w)(pc)	String (1-64)	The name of the storage template volume. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other storage template volumes associated with the parent storage template. Default value: Currently of the form "# GiB *{usage}{index}*", where # is the **size** of the volume, *{usage}* is the **usage**, and *{index}* is a number that may or may not be present to ensure name uniqueness. This form is subject to change in the future.	All
description	(w)(pc)	String (0-100)	Arbitrary text providing additional descriptive information about the template volume. This value will be used to initialize the **description** property of storage volumes that are created using this storage template volume. Default value: An empty string.	All

Table 286. Storage Template Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
size	(w)(pc)	Float	The size in gibibytes (GiB) for this template volume. For volumes in storage templates of **type "fcp"**, the **size** property must be between 1.00 and 1,048,576.00. For volumes in storage templates of **type "fc"**, with a **model** value other than **"EAV"**, the **size** property is fixed at the value specified below and cannot be written: Model - Maximum **size** **"1"** - 0.88 **"2"** - 1.76 **"3"** - 2.64 **"9"** - 7.92 **"27"** - 25.93 **"54"** - 51.86 For volumes with a **model** value of **"EAV"**, **size** is writable and must be between 0.88 and 212489.20. Note however that the maximum is a theoretical size that may not be supported by an operating system. For volumes in storage templates of **type "fc"**, the **size** value is related to the **cylinders** property. Specifying the size in GiB will also define the number of cylinders and vice versa. The **size** and **cylinders** properties cannot be specified together as fields in a create or modify operation. The value will be used to initialize the **size** property of storage volumes that are created using this storage template volume.	All
usage	(w)(pc)	String Enum	The usage of the storage volumes created from this template. Values: • **"boot"** – storage volumes created from this template will contain a bootable image. • **"data"** –storage volumes created from this template will contain data. This value will be used to initialize the usage property of storage volumes that are created using this storage template volume. Default value: **"data"**.	All

Table 286. Storage Template Volume element object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported adapter "type" values**
model	(w)(pc)	String Enum	The 3390 model designation for the storage. Values: **"1"** - Model 1 **"2"** - Model 2 **"3"** - Model 3 **"9"** - Model 9 **"27"** - Model 27 **"54"** - Model 54 **"EAV"** - Extended Address Volume This value will be used to initialize the **model** property of storage volumes that are created using this storage template volume.	fc
cylinders	(w)(pc)	Integer (1113-268434453)	The size of the volume in cylinders. If the **model** value is other than **"EAV"**, the **cylinders** property is fixed at the value specified below and cannot be written: Model - Cylinders **"1"** - 1113 **"2"** - 2226 **"3"** - 3339 **"9"** - 10017 **"27"** - 32760 **"54"** - 65520 For volumes with a **model** value of **"EAV"**, **cylinders** is writable and has a maximum value of 268434453. Note however that this is a theoretical size that may not be supported by an operating system. The **cylinders** value is related to the **size** property. Specifying the size in cylinders will also define the **size** and vice versa. The **cylinders** and **size** properties cannot be specified together as fields in a create or modify operation. This value will be used to initialize the **cylinders** property of storage volumes that are created using this storage template volume.	fc
device-number	(w)(pc)	String (4)	A four-byte lower case hexadecimal string defining the device number that is assigned by default when FICON storage groups containing storage volumes created from this storage template are attached to partitions. This value may be **null**, in which case the system auto-assigns a device number. This value will be used to initialize the **device-number** property of storage volumes that are created using this storage template volume. Default value: **null**.	fc

List Storage Templates

The `List Storage Templates` operation lists the storage templates known to the target Console.

HTTP method and URI

```
GET /api/storage-templates
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property value. Value must be a valid storage template **type** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-templates`	Array of storage-template-info objects	Array of storage-template-info objects, described in the next table. The returned array may be empty.

Each nested storage-template-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Storage Template object.
`cpc-uri`	String/ URI	The **cpc-uri** property of the Storage Template object.
`name`	String	The **name** property of the Storage Template object.
`type`	String Enum	The **type** property of the Storage Template object.

Description

This operation lists the storage templates that are known by the target Console. The object URI, name, type and CPC URI are provided for each.

If the **name** query parameter is specified, the returned list is limited to those storage templates that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid value for the storage template **type** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage templates that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those storage templates that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A storage template is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a storage template, that object is simply omitted from the list but no error status code results.

If no storage templates are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Storage Template object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 642.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 287. List Storage Templates: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-templates HTTP/1.1
x-api-session: 5jb1l6nadc5oggoh5l2xfwo8e1i24y1s6xude47kv4tkasb2ik
```

Figure 326. List Storage Templates: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:44:23 GMT
content-type: application/json;charset=UTF-8
content-length: 366
{
   "storage-templates":[
      {
         "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
         "name":"FICON Template",
         "object-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
         "type":"fc"
      },
      {
         "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
         "name":"FCP Template",
         "object-uri":"/api/storage-templates/6df0fbfa-e215-11e8-bffc-fa163e9c462b",
         "type":"fcp"
      }
   ]
}
```

Figure 327. List Storage Templates: Response

Create Storage Template

The `Create Storage Template` operation creates a new Storage Template object.

HTTP method and URI

```
POST /api/storage-templates
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The value to be set as the storage template's **cpc-uri** property.
`name`	String (1-64)	Required	The value to be set as the storage template's **name** property.
`description`	String (0-200)	Optional	The value to be set as the storage template's **description** property.
`type`	String Enum	Required	The value to be set as the storage template's **type** property.
`shared`	Boolean	Optional	The value to be set as the storage template's **shared** property.
`connectivity`	Integer	Optional	The value to be set as the storage template's **connectivity** property.
`max-partitions`	Integer	Optional	The value to be set as the storage template's **max-partitions** property. The **max-partitions** field is not allowed in the request body unless **the** type field value has a value of **"fcp"**.

Field name	Type	Rqd/Opt	Description
`direct-connection-count`	Integer (0-1000)	Optional	The value to be set as the storage template's **direct-connection-count** property. The **direct-connection-count** field is not allowed in the request body unless the **type** field value has a value of **"fcp"**.
`storage-template-volumes`	Array of storage-template-volume-request-info nested objects	Optional	The set of properties for each of the storage template volumes in the template. An array of one or more storage-template-volume-request-info nested objects, where each element defines the new property values of a storage template volume that is to be created. The **operation** field of each nested object element must be set to **"create"**. For storage templates of **type "fc"**, the storage-template-volume-request-info nested object is defined in Table 246 on page 560. For storage templates of **type "fcp"**, the storage-template-volume-request-info nested object is defined in Table 247 on page 561.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/URI	The **object-uri** property of the newly created Storage Template object.
`element-uris`	Array of String/URI	A list of the URIs for the storage volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the **storage-template-volumes** field in the request body. If no volumes were specified when creating the storage template, the **element-uris** field will be an empty list.

Description

This operation creates a storage template with the values specified and then returns its **object-uri** and the **element-uris** of each storage volume that was created in the response body. The response also includes a Location header that provides the new storage template's URI. Inventory Change notifications for the new storage template and for each new storage template volume identified in the **storage-template-volumes** field are emitted asynchronously to this operation.

If the API user does not have action/task permission to the `Configure Storage – System Programmer` task, a 403 (Forbidden) status code is returned. If the object ID **cpc-uri** field does not identify a CPC object to which the API user has object-access permission, or if the CPC identified by the **cpc-uri** field does not support this operation, a 404 (Not Found) status code is returned. If the CPC identified by the **cpc-uri** field is already associated with a storage template with the specified name, or if two or more of the new storage template volumes have the same name, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM, or if restrictions on the values of the **shared** and **max-partitions** property values, or the **shared** and **direct-connection-count** property values are violated, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage template type. If the request body contents are valid, the storage template and each of the storage template volumes defined in the

storage-template-volumes field are created and their properties are set to their corresponding request body content's field's values. If a field is not found in the request body, its property's value will be defaulted.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 645.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 288. Create Storage Template: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage template with the **name** specified in the request body is already associated with the CPC identified by the **cpc-uri** field.
	15	The **type** field value is **"fc"** and an element of the **storage-template-volumes** field contains both the **size** and **cylinders** fields, or neither the **size** nor **cylinders** fields.
	18	A supplied property is not valid for a storage template's type.
	453	The **name** field for at least two of the nested entries in the **storage-template-volumes** field are the same.
403 (Forbidden)	1	The API user does not have action/task permission to the `Configure Storage - System Programmer` task.
404 (Not Found)	2	The CPC identified by the **cpc-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
	4	The CPC identified by the **cpc-uri** field does not support this operation.
409 (Conflict)	5	The CPC identified by the **cpc-uri** field is not enabled for DPM.
	8	The **max-partitions** field value conflicts with the **shared** field value.
	329	The operation cannot be performed because the CPC designated by the **cpc-uri** is an unmanaged CPC, which is not supported by this operation.
	495	The **direct-connection-count** field value conflicts with the **shared** field value.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-templates HTTP/1.1
x-api-session: 4572v9cswxkse3fgdfxclgakl6c9m67h46jrv06bkgoe42dzdv
content-type: application/json
content-length: 206
{
   "connectivity":4,
   "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
   "name":"FICON Template",
   "storage-template-volumes":[
      {
         "model":"1",
         "operation":"create",
         "usage":"boot"
      }
   ],
   "type":"fc"
}
```

Figure 328. Create Storage Template: Request

```
201 Created
server: Hardware management console API web server / 2.0
location: /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b
cache-control: no-cache
date: Tue, 06 Nov 2018 22:09:21 GMT
content-type: application/json;charset=UTF-8
content-length: 217
{
   "element-uris":[
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/
         9cc736a6-e210-11e8-82d0-fa163e9c462b"
   ],
   "object-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b"
}
```

Figure 329. Create Storage Template: Response

Delete Storage Template

The `Delete Storage Template` operation deletes a storage template.

HTTP method and URI

`DELETE /api/storage-templates/`*`{storage-template-id}`*

In this request, the URI variable *{storage-template-id}* is the object ID of the storage template to delete.

Description

This operation deletes a storage template. All of the template's contained storage template volume elements are also deleted. Inventory Change notifications for the deleted template and element objects are emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission.

If the request is valid, the identified storage template, and all its storage template volumes, are deleted from the CPC.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 289. Delete Storage Template: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage template with the **object-id** *{storage-template-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	2	The storage template object with the **object-id** *{storage-template-id}* was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/storage-templates/6df0fbfa-e215-11e8-bffc-fa163e9c462b HTTP/1.1
x-api-session: 2p68kr4e521yrw0rggj9nad4ax541epqnzsihc67gxqrfpwye6
```

Figure 330. Delete Storage Template: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:56:32 GMT

<No response body>
```

Figure 331. Delete Storage Template: Response

Get Storage Template Properties

The `Get Storage Template Properties` operation retrieves the properties of a single Storage Template object.

HTTP method and URI

```
GET /api/storage-templates/{storage-template-id}
```

In this request, the URI variable *{storage-template-id}* is the object ID of the Storage Template object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Template object as defined in the "Data model" on page 635. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Storage Template object as defined in the "Data model" on page 635.

If the object ID *{storage-template-id}* does not identify a Storage Template object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 649.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 290. Get Storage Template Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A storage template with **object-id** *{storage-template-id}* does not exist on the HMC or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b HTTP/1.1
x-api-session: 61dhyf85cfr0vvn4mt1cxuxbzd1akqx0yizl2kau2vv7e9fjlu
```

Figure 332. Get Storage Template Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:39:09 GMT
content-type: application/json;charset=UTF-8
content-length: 707
{
   "class":"storage-template",
   "connectivity":4,
   "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
   "creation-timestamp":1541542161088,
   "description":"A sample FICON storage template",
   "modification-timestamp":1541543867601,
   "name":"FICON Template",
   "object-id":"9cae8872-e210-11e8-82d0-fa163e9c462b",
   "object-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
   "parent":"/api/console",
   "shared":true,
   "storage-template-volume-uris":[
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/
         9cc736a6-e210-11e8-82d0-fa163e9c462b",
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/
         95da5db0-e214-11e8-b4c0-fa163e9c462b"
   ],
   "type":"fc"
}
```

Figure 333. Get Storage Template Properties: Response

Modify Storage Template Properties

The `Modify Storage Template Properties` operation updates one or more of the writable properties of a storage template.

HTTP method and URI

```
POST /api/storage-templates/{storage-template-id}/operations/modify
```

In this request, the URI variable *{storage-template-id}* is the object ID of the Storage Template object.

Request body contents

Fields for properties whose values are not to be changed by this operation can and should be omitted from the request body.

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Optional	The value to be set as the storage template's **name** property.
`description`	String (0-200)	Optional	The value to be set as the storage template's **description** property.
`shared`	Boolean	Optional	The value to be set as the storage template's **shared** property.

Field name	Type	Rqd/Opt	Description
`connectivity`	Integer	Optional	The value to be set as the storage template's **connectivity** property.
`max-partitions`	Integer	Optional	The value to be set as the storage template’s **max-partitions** property. The **max-partitions** field is not allowed in the request body unless the **type** property has a value of **"fcp"**.
`direct-connection-count`	Integer (0-1000)	Optional	The value to be set as the storage template's **direct-connection-count** property. The **direct-connection-count** field is not allowed in the request body unless the **type** property has a value of **"fcp"**.
`storage-template-volumes`	Array of storage-template-volume-request-info nested objects	Optional	An array of storage-template-volume-request-info nested objects, where each element defines the existing storage template volumes that are to be deleted or the new property values of a storage template volume that is to be created or modified. The storage-template-volume-request-info nested object is defined in the tables that follow. If not specified, the storage template's volumes remain unchanged.

Each nested storage-template-volume-request-info object contains the following fields:

*Table 291. storage-template-volume-request-info nested object for **"create"** operations on **"fc"** storage template volumes*

Field name	Type	Rqd/Opt	Description
`operation`	String Enum	Required	This nested object contains the property values for a new storage template volume that is to be created. Value: **"create"**
`name`	String (1-64)	Optional	The value to be set as the storage template volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage template volume's **description** property.
`size`	Float (0.88-212489.20)	Optional	The value to be set as the storage template volume's **size** property. If the **model** is **"EAV"**, the **size** or **cylinders** fields (but not both) must be specified. If the **model** is not **"EAV"**, the **size** field may not be present in the request body.
`usage`	String Enum	Optional	The value to be set as the storage template volume's **usage** property.
`model`	String Enum	Required	The value to be set as the storage template volume's **model** property.

*Table 291. storage-template-volume-request-info nested object for **"create"** operations on **"fc"** storage template volumes (continued)*			
Field name	**Type**	**Rqd/Opt**	**Description**
`cylinders`	Integer (1113-268434453)	Optional	The value to be set as the storage template volume's **cylinders** property. If the **model** is **"EAV"**, the **size** or **cylinders** fields (but not both) must be specified. If the **model** is not **"EAV"**, the **cylinders** field may not be present in the request body.
`device-number`	String (4)	Optional	The value to be set as the storage template volume's **device-number** property.

*Table 292. storage-template-volume-request-info nested object for **"create"** operations on **"fcp"** storage template volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the property values for a new storage template volume that is to be created. Value: **"create"**
`name`	String (1-64)	Optional	The value to be set as the storage template volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage template volume's **description** property.
`size`	Float (1.00-1048576.00)	Optional	The value to be set as the storage template volume's **size** property.
`usage`	String Enum	Optional	The value to be set as the storage template volume's **usage** property.

*Table 293. storage-template-volume-request-info nested object for **"modify"** operations on **"fc"** storage template volumes*			
Field name	**Type**	**Rqd/Opt**	**Description**
`operation`	String Enum	Required	This nested object contains the new property values for an existing storage template volume that is to be modified. Value: **"modify"**
`element-uri`	String/URI	Required	The canonical URI path for the storage template volume element object that is being updated.
`name`	String (1-64)	Optional	The value to be set as the storage template volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage template volume's **description** property.

*Table 293. storage-template-volume-request-info nested object for **"modify"** operations on **"fc"** storage template volumes (continued)*

Field name	Type	Rqd/Opt	Description
`size`	Float (0.88-212489.20)	Optional	The value to be set as the storage template volume's **size** property. The **size** field is optional but may not be present in the request body if the **cylinders** field is present in the request body or if the **model** value is not **"EAV"**.
`usage`	String Enum	Optional	The value to be set as the storage template volume's **usage** property.
`model`	String Enum	Optional	The value to be set as the storage template volume's **model** property.
`cylinders`	Integer (1113-268434453)	Optional	The value to be set as the storage template volume's **cylinders** property. The **cylinders** field is optional but may not be present in the request body if the **size** field is present in the request body or if the **model** value is not **"EAV"**.
`device-number`	String (4)	Optional	The value to be set as the storage template volume's **device-number** property.

*Table 294. storage-template-volume-request-info nested object for **"modify"** operations on **"fcp"** storage template volumes*

Field name	Type	Rqd/Opt	Description
`operation`	String Enum	Required	This nested object contains the new property values for an existing storage template volume that is to be modified. Value: **"modify"**
`element-uri`	String/ URI	Required	The canonical URI path for the storage template volume element object that is being updated.
`name`	String (1-64)	Optional	The value to be set as the storage template volume's **name** property.
`description`	String (0-100)	Optional	The value to be set as the storage template volume's **description** property.
`size`	Float (1.00-1048576.00)	Optional	The value to be set as the storage template volume's **size** property.
`usage`	String Enum	Optional	The value to be set as the storage template volume's **usage** property.

*Table 295. storage-template-volume-request-info nested object for **"delete"** operations on **"fc"** or **"fcp"** storage template volumes*

Field name	Type	Rqd/Opt	Description
`operation`	String Enum	Required	This nested object identifies an existing storage template volume that is to be deleted. Value: **"delete"**

*Table 295. storage-template-volume-request-info nested object for **"delete"** operations on **"fc"** or **"fcp"** storage template volumes (continued)*			
Field name	**Type**	**Rqd/Opt**	**Description**
`element-uri`	String/ URI	Required	The canonical URI path for the storage template volume element object that is being deleted.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
element-uris	Array of String /URI	A list of the URIs for the storage template volume elements that are created. The order of the URIs in this list will match the order in which the new volumes were specified in the **storage-template-volumes** field in the request body. If the **storage-template-volumes** field did not contain any entries with **operation** equal to **"create"**, the **element-uris** field will be an empty list.

Description

This operation updates a storage template's properties with the values specified and then returns the **element-uris** of each storage template volume that was created in the response body.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a Storage Template object to which the API user has object-access permission, or if the **element-uri** field of a modified or deleted template volume does not exist or is not a member of the storage template.

If the change would put the storage template into a state where its **shared** and **max-partitions** property values or **shared** and **direct-connection-count** property values conflict, or if the a storage template volume modification would put the volume into a state where its **model**, **size** and **cylinders** property values conflict, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the document defines a field that is not supported for the given storage template type, or because the parent CPC is already associated with a storage template with the specified name, or because the operation would put the storage template into a state where two or more of its storage template volumes would have the same name, or because both of, or neither of, the **size** and **cylinders** fields of a EAV FICON storage template volume are defined.

If the request body contents are valid, the storage template's properties are updated to their corresponding request body content's field's values. Optional fields may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified. The element URIs of each new and deleted storage template volume will be added to, or removed from, the storage template's **storage-template-volume-uris** list property.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation. This includes a Property Change notification for the **storage-template-volume-uris** property if the operation creates or deletes storage template volumes.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 654.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 296. Modify Storage Template Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A storage template with the **name** specified in the request body is already associated with the CPC identified by the template's **cpc-uri** property.
	15	The storage template's **type** property value is **"fc"** and an element of the **storage-volumes** field contains both the **size** and **cylinders** fields, or an **operation** value of **"create"** and neither the **size** nor **cylinders** fields.
	18	A supplied property is not valid for a storage template's type.
	453	The operation would put the storage template into a state where the **name** property for at least two of its storage template volumes would be the same.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A storage template with the **object-id** *{storage-template-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	5	An element of the **storage-template-volumes** field has an **element-uri** value that does not exist on the HMC or the API user does not have object-access permission for it, or is not a member of the storage template's **storage-template-volume-uris** array property value.
409 (Conflict)	2	The storage template object with the **object-id** *{storage-template-id}* was busy and the request timed out.
	8	The operation would result in conflicting values for the **max-partitions** and **shared** property values.
	490	The storage template's **type** property value is **"fc"** and an element of the **storage-template-volumes** field would put the storage template volume into a state where its **model**, **size** and **cylinders** property values conflict.
	495	The operation would result in conflicting values for the **direct-connection-count** and **shared** property values.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/operations/modify HTTP/1.1
x-api-session: 1metap37hjlthezzcnd70br2kpn6aoyh97dgxt7rjm7bgmi04z
content-type: application/json
content-length: 376
{
   "description":"A sample FICON storage template",
   "storage-template-volumes":[
      {
         "description":"A Model 3 FICON data volume",
         "model":"3",
         "operation":"create"
      },
      {
         "description":"A Model 1 FICON boot volume",
         "element-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/
            storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
         "operation":"modify"
      }
   ]
}
```

Figure 334. Modify Storage Template Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:37:46 GMT
content-type: application/json;charset=UTF-8
content-length: 142
{
   "element-uris":[
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-
         volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b"
   ]
}
```

Figure 335. Modify Storage Template Properties: Response

List Storage Template Volumes of a Storage Template

The `List Storage Template Volumes of a Storage Template` operation lists the storage template volumes of the storage template with the given identifier.

HTTP method and URI

`GET /api/storage-templates/{storage-template-id}/storage-template-volumes`

In this request, the URI variable *{storage-template-id}* is the **object-id** of the Storage Template object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`maximum-size`	Integer	Optional	Filter value to limit returned object to those that have a **size** property that is less than or equal to **maximum-size**.

Name	Type	Rqd/Opt	Description
`minimum-size`	Integer	Optional	Filter value to limit returned object to those that have a **size** property that is greater than or equal to **minimum-size**.
`usage`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **usage** property. Value must be a valid storage template volume **usage** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`storage-template-volumes`	Array of storage-template-volume-info objects	Array of storage-template-volume-info nested objects, described in the next table. The returned array may be empty.

Each nested storage-template-volume-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path (**element-uri**) of the Storage Template Volume element object.
`name`	String	The **name** property of the storage template volume element.
`size`	Integer	The **size** property of the storage template volume element.
`usage`	String Enum	The **usage** property of the storage template volume element.

Description

This operation lists the storage template volumes that are owned by the identified storage template. The element URI, name, size and usage are provided for each.

If the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

If the **name** query parameter is specified, the returned list is limited to those storage template volumes that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **usage** query parameter is specified, each parameter is validated to ensure it is a valid value for the storage template volume **usage** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those storage template volumes that have a **usage** property matching the specified value. If the **usage** parameter is omitted, this filtering is not done.

If the **minimum-size** query parameter is specified, the returned list is limited to those storage template volumes that have a **size** parameter that is greater than or equal to the specified value. If the **maximum-size** query parameter is specified, the returned list is limited to those storage template volumes that have a **size** parameter that is less than or equal to the specified value. When specified together, the **minimum-size** and **maximum-size** query parameters define a size range on which the volume is filtered.

If either of these query parameters are omitted, the size filter is not bounded on one end. If both of these query parameters are omitted, no filtering on the **size** property is done.

If no storage template volumes are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 657.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 297. List Storage Template Volumes of a Storage Template: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The storage template with the object ID *{storage-template-id}* does not exist on the HMC or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes
   HTTP/1.1
x-api-session: 4po625adh4w8bymp0zc2uaqwjph44bkjh28afrvl4zf7f8qwan
```

Figure 336. List Storage Template Volumes of a Storage Template: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:40:13 GMT
content-type: application/json;charset=UTF-8
content-length: 410
{
   "storage-template-volumes":[
      {
         "element-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/
            storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
         "name":"0.88 GiB Boot",
         "size":0.88,
         "usage":"boot"
      },
      {
         "element-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/
            storage-template-volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b",
         "name":"2.64 GiB Data",
         "size":2.64,
         "usage":"data"
      }
   ]
}
```

Figure 337. List Storage Template Volumes of a Storage Template: Response

Get Storage Template Volume Properties

The `Get Storage Template Volume Properties` operation retrieves the properties of a single Storage Template Volume element object.

HTTP method and URI

```
GET /api/storage-templates/{storage-template-id}/storage-template-volumes/{storage-template-volume-id}
```

In this request, the URI variable *{storage-template-id}* is the object ID of the Storage Template object and the URI variable *{storage-template-volume-id}* is the element ID of the Storage Template Volume element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Storage Template Volume object as defined in the "Data model" on page 635. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the storage template volume object as defined in the "Storage Template Volume element object" on page 638.

A 404 (Not Found) status code is returned if the object ID *{storage-template-id}* does not identify a storage template object to which the API user has object-access permission or if the element ID *{storage-template-volume-id}* does not identify a storage template volume in the storage template.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the storage template whose **object-id** is *{storage-template-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 659.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 298. Get Storage Template Volume Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A storage template with **object-id** *{storage-template-id}* does not exist on the HMC or the API user does not have object-access permission to it.
	5	A storage template volume with **element-id** *{storage-template-volume-id}* does not exist in the storage template on the HMC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-volumes/
   9cc736a6-e210-11e8-82d0-fa163e9c462b HTTP/1.1
x-api-session: 35xby5sl8igqd0pzqxdfjmlad6vlsplfdi2svoeb5sv16qvhco
```

Figure 338. Get Storage Template Volume Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 06 Nov 2018 22:41:23 GMT
content-type: application/json;charset=UTF-8
content-length: 440
{
   "class":"storage-template-volume",
   "cylinders":1113,
   "description":"A Model 1 FICON boot volume",
   "device-number":null,
   "element-id":"9cc736a6-e210-11e8-82d0-fa163e9c462b",
   "element-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/
      storage-template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
   "model":"1",
   "name":"0.88 GiB Boot",
   "parent":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
   "size":0.88,
   "usage":"boot"
}
```

Figure 339. Get Storage Template Volume Properties: Response

Inventory service data

Information about the Storage Templates managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Storage Template objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of **class "storage-template"** are to be included. Information for a particular storage template is included only if the API user has object-access permission to that object.

For each storage template to be included, the inventory response array includes the following:

- An array entry for the storage template object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for "Get Storage Template Properties" on page 649. That is, the data provided is the same as would be provided if a `Get Storage Template Properties` operation were requested targeting this object.
- An array entry for each storage template volume element associated with the storage template. For each such storage volume, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for "Get Storage Template Volume Properties" on page 659.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage template. These objects would appear as multiple array entries in the response array:

```
{
   "class":"storage-template",
   "connectivity":4,
   "cpc-uri":"/api/cpcs/129da68a-05ed-3ae2-b393-3b1442c6c302",
   "creation-timestamp":1541542161088,
   "description":"A sample FICON storage template",
   "modification-timestamp":1541543867601,
   "name":"FICON Template",
   "object-id":"9cae8872-e210-11e8-82d0-fa163e9c462b",
   "object-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
   "parent":"/api/console",
   "shared":true,
   "storage-template-volume-uris":[
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-
         volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
      "/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-template-
         volumes/95da5db0-e214-11e8-b4c0-fa163e9c462b"
   ],
   "type":"fc"
},
{
   "class":"storage-template-volume",
   "cylinders":1113,
   "description":"A Model 1 FICON boot volume",
   "device-number":null,
   "element-id":"9cc736a6-e210-11e8-82d0-fa163e9c462b",
   "element-uri":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b/storage-
      template-volumes/9cc736a6-e210-11e8-82d0-fa163e9c462b",
   "model":"1",
   "name":"0.88 GiB Boot",
   "parent":"/api/storage-templates/9cae8872-e210-11e8-82d0-fa163e9c462b",
   "size":0.88,
   "usage":"boot"
}
```

Figure 340. Storage Template object: Sample inventory data - Response

Tape Library object

A Tape Library object represents a single physical tape storage unit associated with a DPM-enabled CPC. The available pathways to a tape library are defined as instances of Tape Link objects. Tape libraries are automatically discovered, however that discovery requires a single worldwide port name (WWPN) be zoned such that a CPC can detect them. This WWPN is referred to as the management WWPN. The process for supporting tape libraries starts with a request to a storage administrator to

configure the management WWPN (see “Request Tape Library Zoning” on page 670). Once the storage administrator completes that configuration, the CPC is able to discover the tape libraries that are within that network. After the `Request Tape Library Zoning` request is made, library discovery is automatically performed once every 10 minutes. That discovery period changes to once every 24 hours after a tape library has been discovered or 24 hours passes, whichever comes first.

Once they are discovered, tape library instances are persisted and remain visible to an API client even if they are subsequently deconfigured by the storage administrator. If that happens (as indicated by a **"Not Available"** value of the tape library **state** property), it can be removed through the `Undefine Tape Library` operation. Once tape libraries are discovered, links to them are created using “Create Tape Link” on page 685.

Data model

This object includes the properties that are defined in the “Base managed object properties schema” on page 100, with the class-specific specializations identified in Table 300 on page 662. The Tape Library object does not support the operational status related properties.

Table 299. Tape Library object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Tape Library object is of the form `/api/tape-library/{tape-library-id}` where `{tape-library-id}` is the value of the **object-id** property of the Tape Library object.
object-id	—	String (36)	The unique identifier for the tape library instance.
parent	—	String/ URI	The parent of a tape library is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (12)	The class of a Tape Library object is **"tape-library"**.
name	(w)(pc)	String (1-64)	The display name specified for the tape library. The character requirements on this property are the same as those of the **name** property described in the “Base managed object properties schema” on page 100. Names must be unique to the other tape libraries associated to the same CPC.
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the tape library. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 300. Tape Library object: class specific properties

Name	Qualifier	Type	Description
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this tape library object.

Table 300. Tape Library object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
state	(pc)	String Enum	The visibility of this tape library. Values: • **"available"** – The tape library can be sensed on at least one physical path. • **"not-available"** – The tape library cannot be sensed on any path.
vendor-id	—	String (1-8)	The tape library vendor.
device-id	—	String	An identifier assigned by the vendor that uniquely identifies this tape library. It is suggested, but not mandated, that this field include a model and serial number.

List Tape Libraries

The `List Tape Libraries` operation lists the tape libraries known to the target Console.

HTTP method and URI

```
GET /api/tape-libraries
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`state`	String Enum	Optional	Filter string to limit the returned objects to those that have a matching **state** property. Value must be a valid tape library **state** property value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`tape-libraries`	Array of tape-library-info objects	Array of tape-library-info objects, described in the next table. The returned array may be empty.

Each nested tape-library-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Tape Library object.

Field name	Type	Description
`name`	String	The **name** property of the Tape Library object.
`cpc-uri`	String/ URI	The **cpc-uri** property of the Tape Library object.
`state`	String Enum	The **state** property of the Tape Library object.

Description

This operation lists the tape libraries that are known to the target Console. The object URI, name, state and the URI of its associated CPC are provided for each.

If the **name** query parameter is specified, the returned list is limited to those tape libraries that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **state** query parameter is specified, the parameter is validated to ensure it is a valid value for the tape library **state** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those tape libraries that have a **state** property matching the specified value. If the **state** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those tape libraries that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A tape library is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a tape library, that object is simply omitted from the list but no error status code results.

If no tape libraries are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Tape Library object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 663.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 301. List Tape Libraries: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-libraries HTTP/1.1
x-api-session: 58zctc8jvhdh9hd8dji9wvuoh80t0l2rdk4bvrp95rmht2o8tq
```

Figure 341. List Tape Libraries: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 20 Jan 2021 16:04:00 GMT
content-type: application/json;charset=UTF-8
content-length: 207
{
   "tape-libraries":[
      {
         "cpc-uri":"/api/cpcs/406a206e-e4ae-3277-bcc2-30c136208dd9",
         "name":"3573-TL 00L2U78Z8185_LL0",
         "object-uri":"/api/tape-libraries/031e2350-5b37-11eb-b81d-fa163e11e9ec",
         "state":"available"
      }
   ]
}
```

Figure 342. List Tape Libraries: Response

Undefine Tape Library

The `Undefine Tape Library` request removes a deconfigured tape library from the list of tape libraries associated with a CPC.

HTTP method and URI

```
POST /api/tape-libraries/{tape-library-id}/operations/undefine
```

In this request, the URI variable *{tape-library-id}* is the object ID of the Tape Library object.

Description

This operation removes a single tape library from the persisted list of tape libraries known to a CPC. Only tape libraries that are not referenced by any tape link and tape libraries that are no longer configured in the network, as indicated by a **"not-available"** value of the tape library's **status** property, can be deleted.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape library whose **object-id** is *{tape-library-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 302. Undefine Tape Library: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	1	A tape library with the **object-id** *{tape-library-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	1	The value of the **state** property for the tape library with the **object-id** *{tape-library-id}* is not **"not-available"**.
	2	The tape library object with the **object-id** *{tape-library-id}* was busy and the request timed out.
	446	The tape library with the **object-id** *{tape-library-id}* has at least one tape link that references it.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/tape-libraries/031e2350-5b37-11eb-b81d-fa163e11e9ec/operations/undefine HTTP/1.1
x-api-session: 51kgglqa8dwfnurheeuypzk2vcl51ajdl3hezpfwasi7q67liq
content-type: application/json
```

Figure 343. Undefine Tape Library: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 20 Jan 2021 16:10:23 GMT

<No response body>
```

Figure 344. Undefine Tape Library: Response

Get Tape Library Properties

The `Get Tape Library Properties` request returns the complete set of properties defined for a single tape library.

HTTP method and URI

```
GET /api/tape-libraries/{tape-library-id}
```

In this request, the URI variable *{tape-library-id}* is the object ID of the Tape Library object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Tape Library object as defined in the "Data model" on page 662. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Tape Library object as defined in the "Data model" on page 662.

If the object ID *{tape-library-id}* does not identify a Tape Library object on the Console, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape library whose **object-id** is *{tape-library-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 667.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 303. Get Tape Library Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A tape library with **object-id** *{tape-library-id}* does not exist on the Console or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-libraries/031e2350-5b37-11eb-b81d-fa163e11e9ec HTTP/1.1
x-api-session: 2bwrdbymnco1dr84guh8fogvi7blnj1y25x0yppkz9lb5h8izo
```

Figure 345. Get Tape Library Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 20 Jan 2021 16:05:43 GMT
content-type: application/json;charset=UTF-8
content-length: 358
{
   "class":"tape-library",
   "cpc-uri":"/api/cpcs/406a206e-e4ae-3277-bcc2-30c136208dd9",
   "description":"",
   "device-id":"3573-TL 00L2U78Z8185_LL0",
   "name":"3573-TL 00L2U78Z8185_LL0",
   "object-id":"031e2350-5b37-11eb-b81d-fa163e11e9ec",
   "object-uri":"/api/tape-libraries/031e2350-5b37-11eb-b81d-fa163e11e9ec",
   "parent":"/api/console",
   "state":"available",
   "vendor-id":"IND"
}
```

Figure 346. Get Tape Library Properties: Response

Update Tape Library Properties

The `Update Tape Library Properties` operation updates one or more of the writable properties of a tape library.

HTTP method and URI

```
POST /api/tape-libraries/{tape-library-id}
```

In this request, the URI variable *{tape-library-id}* is the object ID of the Tape Library object.

Request body contents

The request body is expected to contain a JSON object that provides the new value of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Data model" on page 662. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a tape library's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-library-id}* does not identify a tape library object on the Console.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because another tape library with the specified name exists within the same associated CPC.

If the request body contents are valid, the tape library's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape library whose **object-id** is *{tape-library-id}*.

- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 304. Update Tape Library Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A tape library with the **name** specified in the request body is already associated to the CPC identified by its **cpc-uri** property.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	The tape library with the **object-id** *{tape-library-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	2	The Tape Library object with the **object-id** *{tape-library-id}* was busy and the request timed out.
503 (Service Unavailable	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/tape-libraries/efe36382-50f9-11eb-aacc-00106f23d200 HTTP/1.1
x-api-session: 2k645hispo2mmy4gtaw06yfyk78a9ixh7bjr18t4lndnjfex5v
content-type: application/json
content-length: 65
{
   "description":"Used For Data",
   "name":"Tape Library For Data"
}
```

Figure 347. Update Tape Library Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 15 Jan 2021 10:11:07 GMT

<No response body>
```

Figure 348. Update Tape Library Properties: Response

Request Tape Library Zoning

The `Request Tape Library Zoning` operation sends a request to the storage administrator to set up the zoning so that tape libraries can be discovered.

HTTP method and URI

```
POST /api/tape-libraries/operations/request-tape-library-zoning
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The canonical URI path of the CPC object that is to be configured.
`email-to-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be notified through email of the storage resources that require configuration to support tape library discovery. These email addresses will appear in the "to:" address list in the email that is sent. The **email-cc-addresses** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. An email will not be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the storage resources that require configuration to support tape library discovery. These email addresses will appear in the "cc:" address list in the email that is sent. Default value: **null**. No one will be copied on the email.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`management-world-wide-port-name`	String (16)	A 16-character lower case string that contains the management World Wide Port Name (WWPN) that needs to be configured in the SAN so that tape libraries can be discovered. This value is also available in the **management-world-wide-port-name** property of the CPC object.

Description

The Request Tape Library Zoning requests generates a worldwide port name (WWPN) and optionally sends it to a storage administrator with a request that it be zoned so that a CPC can use it to discover tape libraries in the network. The first time this request is issued for a CPC, a management WWPN is generated and returned in the request body. If so requested, emails are sent with the information a storage administrator needs to configure that WWPN in the SAN. If this request is issued again targeting the same CPC, a new WWPN is not generated. The previously generated WWPN is returned, and if requested, the emails are resent.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the value of the **cpc-uri** field does not identify a CPC object to which the API user has object-access permission.

If the CPC identified by **cpc-uri** is not active, or if it is not a DPM CPC with the **dpm-fcp-tape-management** feature enabled, or is an unmanaged CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the **email-cc-addresses** field is present in the request body without the **email-to-addresses** field, or because any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, and this is the first time a request for zoning has been made, a WWPN is generated and set as the value of the CPC's **management-world-wide-port-name** property. An email containing information about the WWPN that the storage administrator is to configure is optionally sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the Console is not configured to support emails. A failure to send the email does not roll back the creation of the management WWPN. An API client should assume that a management WWPN was created even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be re-sent by issuing another `Request Tape Library Zoning` request.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 670.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 305. Request Tape Library Zoning: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A CPC with the **object-uri cpc-uri** does not exist on the Console or the API user does not have object-access permission for it.

Table 305. Request Tape Library Zoning: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	5	The CPC with the **object-uri cpc-uri** is not enabled for DPM.
	13	The CPC with the **object-uri cpc uri** does not support the **dpm-fcptape-management** feature.
	329	The operation cannot be performed because the CPC with the **object-uri cpc-uri** is an unmanaged CPC, which is not supported by this operation.
	487	The CPC with the **object-uri cpc-uri** does not have any adapters configured as FCP.
	491	An error occurred when sending the email.
	493	The **email-to-addresses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-libraries/operations/request-tape-library-zoning HTTP/1.1
x-api-session: 2hryzzbabcrgbtwdjiskzim2lbyapybm2t60p211cnrihlf6po
content-type: application/json
content-length: 61
{
   "cpc-uri":"/api/cpcs/6b7b1607-6d90-3366-b9f7-1a0b3e9b8e16"
}
```

Figure 349. Request Tape Library Zoning: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 22 Jan 2021 09:12:52 GMT
content-type: application/json;charset=UTF-8
content-length: 54
{
   "management-world-wide-port-name":"a1b2c3d4e5f60007"
}
```

Figure 350. Request Tape Library Zoning: Response

Discover Tape Libraries

The `Discover Tape Libraries` operation triggers asynchronous discovery of tape libraries. Because the system periodically performs discovery for tape libraries, it is not necessary to use this operation;

however, issuing this operation may cause the discovery to be performed sooner than it would have otherwise.

HTTP method and URI

```
POST /api/tape-libraries/operations/discover-tape-libraries
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The canonical URI path of the CPC object from which discovery is performed.
`force-restart`	Boolean	Optional	Indicates if there is an in-progress discovery operation for the target CPC, it should be terminated and started again. If **false** and there is an in-progress discovery operation for the target CPC, no new discovery is started. Default value: **false**

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 675. The **job-results** field is **null** when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

Requests that a discovery of tape libraries visible to a specific CPC is performed immediately. Normally, tape library discovery is performed automatically once every 24 hours. This request can be used to force the tape library discovery process to start immediately.

Discovery requires that the management worldwide port name (WWPN) has been generated for the target CPC using the Request Tape Library Zoning operation, and that WWPN has been configured in the SAN.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the value of the **cpc-uri** field does not identify a CPC object to which the API user has object-access permission.

If the CPC identified by **cpc-uri** is not active, or if it is not a DPM CPC with the **dpm-fcp-tape-management** feature enabled, or is an unmanaged CPC, or if it has not been zoned, a 409 (Conflict) status code is returned.

If the request body contents are valid, the CPC initiates a discovery of tape libraries. When the asynchronous job completes, a subsequent `List Tape Libraries` operation will contain the complete list of the tape libraries that were discovered.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 673.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 306. Discover Tape Libraries: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks.
404 (Not Found)	2	A CPC with the **object-uri cpc-uri** does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	5	The CPC with the **object-uri cpc-uri** is not enabled for DPM.
	13	The CPC with the **object-uri cpc-uri** does not support the **dpm-fcp-tape-management** feature
	329	The operation cannot be performed because the CPC identified by **cpc-uri** is an unmanaged CPC, which is not supported by this operation.
	501	The CPC with the **object-uri cpc-uri** has not been zoned.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 307. Discover Tape Libraries: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	The operation completed successfully.
500 (Server Error)	263	The library discovery operation failed and there will be no automatic retry.

Example HTTP interaction

```
POST /api/tape-libraries/operations/discover-tape-libraries HTTP/1.1
x-api-session: 5ntw2uwfy93uzz1ffut1n1yqvj6x8mvg9pnuygpz0zp9ju16bl
content-type: application/json
content-length: 61
{
   "cpc-uri":"/api/cpcs/6b7b1607-6d90-3366-b9f7-1a0b3e9b8e16"
}
```

Figure 351. Discover Tape Libraries: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/03b73154-5c94-11eb-b398-fa163e0dff18
cache-control: no-cache
date: Fri, 22 Jan 2021 09:27:15 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/03b73154-5c94-11eb-b398-fa163e0dff18"
}
```

Figure 352. Discover Tape Libraries: Response

Inventory service data

Information about the tape libraries managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Tape Library objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of **class "tape-library"** are to be included. Information for a particular tape library is included only if the API user has object-access permission to that object.

For each tape library to be included, the inventory response array includes the following:

- An array entry for the tape library object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for “Get Tape Library Properties” on page 666. That is, the data provided is the same as would be provided if a `Get Tape Library Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage template. These objects would appear as multiple array entries in the response array:

```
The following fragment is an example of the JSON objects that would be included in the Get
Inventory
response to describe a tape library. These objects would appear as multiple array entries in the
response array:
{
   "class":"tape-library",
   "cpc-uri":"/api/cpcs/d629c0a5-80cf-3590-b7f0-2456edc6652d",
   "description":"",
   "device-id":"3573-TL 00L2U78Z8185_LL0",
   "name":"3573-TL 00L2U78Z8185_LL0",
   "object-id":"4f3400a4-6661-11eb-86b8-fa163e65c83b",
   "object-uri":"/api/tape-libraries/4f3400a4-6661-11eb-86b8-fa163e65c83b",
   "parent":"/api/console",
   "state":"available",
   "vendor-id":"IND"
}
```

Figure 353. Tape Library object: Sample inventory data - Response

Tape Link object

A Tape Link object represents a single tape link associated with a DPM-enabled CPC. Tape links define pathways to tape library storage that can be attached to partitions. When a tape link is attached to a partition, its fulfilled resources are virtualized and the partition's view of them is represented by a set of Virtual Tape Resource element objects. The Tape Link object APIs provide access to the set of tape links that are associated with a CPC that is enabled for DPM. APIs exist to create and delete tape links, and to query and modify tape link properties. APIs also exist to query and update selected properties of the virtual tape resource elements of tape links.

A tape link transitions through a number of states in its lifecycle. It is complicated by the fact that some of its attributes cannot be realized without changes to the configuration of the Storage Area Network (SAN) in which the storage resources defined in the tape link reside. Creation or modification of a tape link by a system administrator requires a subsequent fulfillment action by the SAN administrator before the storage resources in that tape link can be used by a partition. Fulfillment of tape link resources is auto-detected.

A tape link has a **fulfillment-state** property that indicates a tape link's current fulfillment state. The following table lists important steps in a tape link's lifecycle, and their effect on the tape link's **fulfillment-state** and other properties, and on the partitions to which they are attached.

Table 308.

Lifecycle step	Comment
A tape link is created.	The **fulfillment-state** property of the tape link is set to **"pending"**. An email is optionally sent to the SAN administrator requesting fulfillment of the new tape link's storage resources. The tape link may be attached to a partition, but if activated, the tape link's storage resources will not be available to the partition.

Table 308. (continued)	
Lifecycle step	**Comment**
The tape link is fulfilled.	The **fulfillment-state** property of the tape link is set to **"complete"**. The tape link's storage resources are now available to a partition. If the tape link is currently attached to an active partition, the resources will be dynamically made available to the partition. Note that there is no guarantee that all of a tape link's resources will be fulfilled at the same time, so it is possible for a tape link to be in a state of partial fulfillment. In that case, some of the resources will have been fulfilled, and some have not. The tape link's **fulfillment-state** property will be updated to **"complete"** only after all its resources have been fulfilled.
The tape link is modified.	If a tape link is modified to request an increase in the connectivity or the maximum number of partitions to which it can be attached, the tape link **fulfillment-state** property is set to **"pending"**. If the tape link is attached to a partition that is currently active or subsequently activated, it will continue to see the original storage resources. If the modification requires new resources, or frees up existing resources, an email is optionally sent to the SAN administrator requesting fulfillment of the new resources or deletion of unneeded resources.
The tape link's new resources are fulfilled.	The **fulfillment-state** property of the tape link is set to **"complete"**. The tape link's new storage resources are now available to a partition. If attached to an active partition, its resources will be dynamically changed.
The tape link is attached to a partition.	If the partition is currently active, the tape link's fulfilled resources are dynamically added to the partition. A tape link can be attached to a partition any time after it is created.
The tape link is detached from a partition.	If the partition is currently active, the tape link's storage resources are dynamically removed from the partition.
The tape link is deleted.	The tape link must be detached from all partitions to which it is attached before it can be deleted.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the class-specific specializations identified in Table 310 on page 678. The Tape Link object does not support the operational status related properties.

Table 309. Tape Link object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Tape Link object is of the form `/api/tape-links/{tape-link-id}` where `{tape-link-id}` is the value of the **object-id** property of the Tape Link object.
object-id	—	String (36)	The unique identifier for the tape link instance.
parent	—	String/ URI	The parent of a tape library is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (9)	The class of a Tape Link object is **"tape-link"**.

Table 309. Tape Link object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the tape link. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other tape links associated to the same CPC.
description	(w)(pc)	String (0-200)	Arbitrary text providing additional descriptive information about the tape link. Default value: An empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 310. Tape Link object: class specific properties			
Name	**Qualifier**	**Type**	**Description**
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this Tape Link object.
fulfillment-state	(pc)	String Enum	The current fulfillment state of the tape link. Values: • **"complete"** – All resources in this tape link have been fulfilled. • **"pending"** – The tape link has been created or modified, but its resources have not yet been completely fulfilled by the SAN administrator. • **"pending-with-mismatches"** – One or more adapters with visibility to the tape link, but were not requested, have been discovered. • **"incomplete"** – At least one backing adapter is in an error state, or an error or an unrequested reconfiguration in the SAN that impacts requested resources, or an unrequested tape library is discovered, or an additional (unrequested) tape library is discovered, or no tape libraries are discovered. The **incomplete-reasons** property provides additional information on the reason why the tape link is in this state. Default value: **"pending"**

Table 310. Tape Link object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
virtual-tape-resource-uris	(c)(pc)	Array of String/ URI	The list of virtual tape resources for this tape link. Each element in this array is the canonical URI path of a Virtual Tape Resource element object. This value will change, and property change notification emitted, when the tape link has fulfilled resources and is attached to or detached from partitions, or when the tape link is already attached to a partition and resources are fulfilled again due to a change in the **connectivity** or **max-partitions** properties. Default value: An empty array
tape-library-uri	(pc)	String/ URI	The canonical URI path of the Tape Library object linked by this Tape Link object. This value will be **null**, if a tape library was not specified when creating the tape link and **fulfillment-state** is not **"complete"**. Default value: **null**, indicating the tape library will be chosen by the storage administrator.
adapter-port-uris	(c)(pc)	Array of String/ URI	The list of requested storage adapter ports for this tape link. Each element in this array is the canonical URI path of an Adapter Port element object. The size of the **adapter-port-uris** array cannot be greater than the value of the **connectivity** property. This value will change as storage adapter ports are added and removed using the `Add Adapter Ports` or `Remove Adapter Ports` operations. Default value: An empty array If a tape link is created without specifying adapter ports, or fewer adapter ports than the value of **connectivity**, the remaining ports are assigned as follows: • If **tape-library-uri** is specified when the tape link is created, the matching adapters for that tape library will be assigned to the new tape link. • If **tape-library-uri** is not specified, the adapters will be chosen by the storage administrator and the number of adapter ports referenced in this property will be smaller than **connectivity** until that happens.

Table 310. Tape Link object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
connectivity	(w)(pc)	Integer (1-255)	The number of adapters to utilize for this tape link. The maximum value is the lesser of **255** or the number of ports on FICON adapters of **type "fcp"**. The value of **connectivity** cannot be smaller than the number of adapter ports referenced in the **adapter-port-uris** property. Default value: 2
max-partitions	(w)(pc)	Integer	The maximum number of partitions to which this tape link can be attached. The value of **max-partitions** cannot be decreased to a value that is less that the total number of partitions to which this tape link is currently attached. The minimum value for the **max-partitions** property is **1**; the maximum is the value of the CPC object's **maximum-partitions** property. Default value: 1
unassigned-world-wide-port-names	(c)(pc)	Array of world-wide-port-name-info	The list of information about the worldwide port names (WWPNs) that have been allocated to support this tape link, but have not yet been assigned to a virtual tape resource. Each element in this array is an instance of a world-wide-port-name-info nested object, defined in Table 240 on page 544. This value will change, and property change notifications emitted, when the **connectivity** or **max-partitions** properties change, or when virtual tape resources are created or deleted. Default value: An empty array

Table 310. Tape Link object: class specific properties (continued)			
Name	**Qualifier**	**Type**	**Description**
incomplete-reasons	—	Array of String Enum	The list of reasons for the tape link being in **"incomplete"** state. Values: • **"zoning-error"**- There is an error with WWPN zoning. • **"adapter-degraded"**– An adapter is degraded. • **"adapter-link-error"**– Cable or optics errors were detected for the physical connection to the fabric. • **"requested-library-not-discovered"** - Configuration of the requested library is pending. • **"multiple-libraries-discovered"**- More than one library was discovered for the requested tape link. • **"unknown-library-discovered"**- Drives were sensed that could not be associated with a library. • **"different-library-discovered"**- A tape library was discovered that was not requested for the tape link. • **"library-not-reachable"** - The tape library can no longer be reached due to an error with cables or with the library itself. This field is present only when the value of the **fulfillment-state** of the tape link is **"incomplete"**.

Virtual Tape Resource element object

A virtual tape resource defines the virtualized view of a storage adapter that is backing a tape link as seen by a partition.

Table 311. Tape Link object - Virtual Tape Resource element object properties			
Name	**Qualifier**	**Type**	**Description of specialization**
element-uri	—	String/ URI	The canonical URI path for the Virtual Tape Resource element object is of the form `/api/tape-link/{tape-link-id}/virtual-tape-resources/{virtual-tape-resource-id}`, where *{tape-link-id}* is the **object-id** of the containing tape link, and *{virtual-tape-resource-id}* is the **element-id** of this virtual tape resource.
element-id	—	String (36)	The unique identifier for the virtual tape resource element instance.
parent	—	String/ URI	The parent of a virtual tape resource is its owning tape link, so the parent value is the canonical URI path for the tape link.
class	—	String (21)	The class of a Virtual Tape Resource element object is **"virtual-tape-resource"**.

Table 311. Tape Link object - Virtual Tape Resource element object properties (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the virtual tape resource. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other virtual tape resources within the parent tape link.
description	(w)(pc)	String (0-1024)	Arbitrary text providing additional descriptive information about the virtual tape resource. Default value: an empty string.
partition-uri	—	String/ URI	The canonical URI path of the partition to which this virtual tape resource is attached.
device-number	(w)(pc)	String (4)	Device number of the virtual tape resource. The value must be a 4-digit lower case hexadecimal. The value must be unique across the device numbers of all other virtual tape resource elements and all instances of the objects listed in "Channel-based device numbers" on page 197 of the partition identified by **partition-uri**. Default value: Auto-generated
adapter-port-uri	(pc)	String/ URI	The canonical URI path for the backing adapter port. The **type** field of the parent adapter must be **"fcp"**. This value will change when the adapter ports are added or removed through the `Add Adapter Ports`, `Remove Adapter Ports`, or `Replace Adapter Port` operations. The value of this property will be **null** if an adapter has not been discovered to back this virtual tape resource.
world-wide-port-name-info	—	world-wide-port-name-info object	Information about the worldwide port name allocated to the virtual tape resource. The element is an instance of a world-wide-port-name-info nested object, as described in Table 240 on page 544.
degraded-reasons	(pc)	Array of String Enum	The list of reasons for the degradation of the virtual tape resource. One or more of the following: • **"adapter"** - The status of the backing adapter of the virtual tape resource is either **"service"** or **"exceptions"**. • **"storage-configuration"** - The virtual tape resource is not able to access the tape library. If the status of the partition that this virtual tape resource is associated with is neither **"degraded"** nor **"reservation-error"**, this list will be empty.

List Tape Links

The `List Tape Links` operation lists the tape links known to the target Console.

HTTP method and URI

```
GET /api/tape-links
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`fulfillment-state`	String Enum	Optional	Filter string to limit the returned objects to those that have a matching **fulfillment-state** property. Value must be a valid tape link **fulfillment-state** property value.
`tape-library-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **tape-library-uri** property. An empty **tape-library-uri** query value can be specified to match tape links that have a **null tape-library-uri** value.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`tape-links`	Array of tape-link-info objects	Array of tape-link-info objects, described in the next table. The returned array may be empty.

Each nested tape-link-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path (**object-uri**) of the Tape Link object.
`cpc-uri`	String/ URI	The **cpc-uri** property of the Tape Link object.
`name`	String	The **name** property of the Tape Link object.
`fulfillment-state`	String Enum	The **fulfillment-state** property of the Tape Link object.
`tape-library-uri`	String/ URI	The **tape-library-uri** property of the Tape Link object.

Description

This operation lists the tape links that are known to the target Console. The object URI, name, fulfillment state and the URIs of its associated CPC and tape library are provided for each.

If the **name** query parameter is specified, the returned list is limited to those tape links that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **fulfillment-state** query parameter is specified, the parameter is validated to ensure it is a valid value for the tape link **fulfillment-state** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those tape links that have a **fulfillment-state** property matching the specified value. If the **fulfillment-state** parameter is omitted, this filtering is not done.

If the **cpc-uri** or **tape-library-uri** query parameters are specified, the returned list is limited to those tape links that have a matching **cpc-uri** or **tape-library-uri** property. If the **cpc-uri** or **tape-library-uri** parameters are omitted, this filtering is not done for the missing parameter.

A tape link is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a tape link, that object is simply omitted from the list but no error status code results.

If no tape links are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to each Tape Link object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 683.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 312. List Tape Links: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links HTTP/1.1
x-api-session: 28fgd89myas89kwzak2tkqrcz0squ3e6wogd8nipmlo5wzjz4t
```

Figure 354. List Tape Links: Request

```
200 OK
 server: Hardware management console API web server / 2.0
 cache-control: no-cache
 date: Wed, 09 Dec 2020 04:46:39 GMT
 content-type: application/json;charset=UTF-8
 content-length: 273
 {
    "tape-links":[
       {
          "cpc-uri":"/api/cpcs/c4d14a49-1277-3d8b-8b3e-4be788e648a8",
          "fulfillment-state":"complete",
          "name":"Tape Link",
          "object-uri":"/api/tape-links/268047d0-39d9-11eb-b329-fa163e874820",
          "tape-library-uri":"/api/tape-libraries/2933d5be-39d9-11eb-9fc7-fa163e874820"
       }
    ]
 }
```

Figure 355. List Tape Links: Response

Create Tape Link

The `Create Tape Link` operation creates a new tape link object.

HTTP method and URI

```
POST /api/tape-links
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Required	The value to be set as the tape link's **cpc-uri** property.
`name`	String (1-64)	Required	The value to be set as the tape link's **name** property.
`description`	String (1-200)	Optional	The value to be set as the tape link's **description** property.
`max-partitions`	Integer	Optional	The value to be set as the tape link's **max-partitions** property.
`connectivity`	Integer	Optional	The value to be set as the tape link's **connectivity** property.
`tape-library-uri`	String/ URI	Optional	The value to be set as the tape link's **tape-library-uri** property.
`adapter-port-uris`	Array of String/ URI	Optional	The value to be set as the tape link's **adapter-port-uris** property. If the **adapter-port-uris** field is omitted or has a size less than the value of **connectivity**, the storage administrator is expected to pick the remaining adapter ports that are to be assigned to this tape link.

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Optional	A set of one or more email addresses for the people that are to be notified through email of the new tape link resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. Default value: **null**. No email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the new tape link resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the new tape link resources that require fulfillment. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. An email without a special text insert will be sent.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the newly created Tape Link object.

Description

This operation creates a tape link with the values specified and then returns its **object-uri**. The response also includes a Location header that provides the new tape link's URI. An Inventory Change notification for the new tape link is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. If the **cpc-uri** field does not identify a CPC object to which the API user has object-access permission, a 404 (Not Found) status code is returned. A 404 (Not Found) status code is also returned if the **tape-library-uri** field does not identify a Tape Library object to which the API user has object-access permission, or any member of the **adapter-port-uris** field does not identify an Adapter Port object to which the API user has object-access permission. If the CPC identified by the **cpc-uri** field is already associated with a tape link with the specified name, or if the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned. If the CPC identified by the **cpc-uri** field is not enabled for DPM or does not have the **dpm-fcp-tape-management** feature enabled or is not active, or if no FCP adapters are configured, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. If the request body contents are valid, the tape link is created and its properties are set to their corresponding request body

content's field's values. If a field is not found in the request body, its property's value will be defaulted. The **fulfillment-state** property of the tape link is set to **"pending"** indicating action is required by the SAN administrator.

If the **email-to-addresses** field is present and not null in the request body, an email containing information about the tape link resources that require fulfillment is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not null, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the HMC is not configured to support emails. A failure to send the email does not rollback the creation of the tape link. An API client should assume that a tape link was created even though the request failed with a 409 (Conflict) status code and 491 reason code. The URI of the new Tape Link object is returned in the **error-details** field in the response body. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the `Resend Request` operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is **cpc-uri**.
- Object-access permission to the tape library whose **object-id** is *{tape-library-id}*.
- Object-access permission to each adapter referenced in **adapter-port-uris**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 686.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 313. Create Tape Link: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A tape link with the **name** specified in the request body is already associated with the CPC identified by the **cpc-uri** specified in the request body.
	15	The number of adapters specified in **adapter-port-uris** is larger than connectivity.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.

Table 313. Create Tape Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	2	A CPC with the **object-uri cpc-uri** does not exist on the Console or the API user does not have object-access permission for it.
	442	At least one adapter port referenced in **adapter-port-uris** does not exist on the Console or the API user does not have object-access permission for it.
	447	A tape library with the **object-id** *{tape-library-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	5	The CPC identified by the **cpc-uri** field is not enabled for DPM.
	13	The CPC identified by the **cpc-uri** field does not support the **dpm-fcp-tape-management** feature.
	329	The operation cannot be performed because the CPC identified by **cpc-uri** is an unmanaged CPC, which is not supported by this operation.
	483	The **type** property value for at least one adapter referenced in the **adapter-port-uris** field is not **"fcp"**.
	487	The CPC with the **object-uri cpc-uri** does not have any adapters configured as FCP.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, a new tape link will have been created.
	493	The **email-to-addresses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-links HTTP/1.1
 x-api-session: gi7oe4ef8z2vlgjju4yvxdymnh1q2r8xh3zp47ddssyamxo5p
 content-type: application/json
 content-length: 235
 {
    "adapter-port-uris":[
       "/api/adapters/3fc9b166-3995-11eb-befd-fa163e874820/storage-ports/0"
    ],
    "cpc-uri":"/api/cpcs/c4d14a49-1277-3d8b-8b3e-4be788e648a8",
    "description":"An example tape link",
    "max-partitions":4,
    "name":"Tape Link"
 }
```

Figure 356. Create Tape Link: Request

```
201 Created
 server: Hardware management console API web server / 2.0
 location: /api/tape-links/268047d0-39d9-11eb-b329-fa163e874820
 cache-control: no-cache
 date: Wed, 09 Dec 2020 04:43:59 GMT
 content-type: application/json;charset=UTF-8
 content-length: 69
 {
    "object-uri":"/api/tape-links/268047d0-39d9-11eb-b329-fa163e874820"
 }
```

Figure 357. Create Tape Link: Response

Get Tape Link Properties

The `Get Tape Link Properties` operation returns the complete set of properties defined for a single tape link.

HTTP method and URI

```
GET /api/tape-links/{tape-link-id}
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Tape Link object as defined in the "Data model" on page 677. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Tape Link object as defined in the "Data model" on page 677.

If the object ID *{tape-link-id}* does not identify a Tape Link object on the Console, a 404 (Not Found) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 689.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 314. Get Tape Link Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A tape link with **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links/268047d0-39d9-11eb-b329-fa163e874820 HTTP/1.1x-api-session:
3ezz3pq40rmeka7k6cf6ct0h6y67lgs64op856vrxusuut235v
```

Figure 358. Get Tape Link Properties: Request

```
200 OK
 server: Hardware management console API web server / 2.0
 cache-control: no-cache
 date: Wed, 09 Dec 2020 04:48:23 GMT
 content-type: application/json;charset=UTF-8
 content-length: 844
 {
    "adapter-port-uris":[
       "/api/adapters/3fc9b166-3995-11eb-befd-fa163e874820/storage-ports/0"
    ],
    "class":"tape-link",
    "connectivity":1,
    "cpc-uri":"/api/cpcs/c4d14a49-1277-3d8b-8b3e-4be788e648a8",
    "description":"An example tape link",
    "fulfillment-state":"complete",
    "max-partitions":4,
    "name":"Tape Link",
    "object-id":"268047d0-39d9-11eb-b329-fa163e874820",
    "object-uri":"/api/tape-links/268047d0-39d9-11eb-b329-fa163e874820",
    "parent":"/api/console",
    "tape-library-uri":"/api/tape-libraries/2933d5be-39d9-11eb-9fc7-fa163e874820",
    "unassigned-world-wide-port-names":[
       {
          "status":"validated",
          "world-wide-port-name":"a1b2c3d4e5f601d5"
       },
       {
          "status":"validated",
          "world-wide-port-name":"a1b2c3d4e5f601d6"
       },
       {
          "status":"validated",
          "world-wide-port-name":"a1b2c3d4e5f601d3"
       },
       {
          "status":"validated",
          "world-wide-port-name":"a1b2c3d4e5f601d7"
       }
    ],
    "virtual-tape-resource-uris":[]
 }
```

Figure 359. Get Tape Link Properties: Response

Modify Tape Link Properties

The `Modify Tape Link Properties` operation updates one or more of the writable properties of a tape link.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/modify
```

In this request, the URI variable *{tape link-id}* is the object ID of the Tape Link object.

Request body contents

Fields for properties whose values are not to be changed by this operation can and should be omitted from the request body.

Field name	Type	Rqd/Opt	Description
`name`	String (1-64)	Optional	The value to be set as the tape link's **name** property.
`description`	String (1-200)	Optional	The value to be set as the tape link's **description** property.

Field name	Type	Rqd/Opt	Description
`max-partitions`	Integer	Optional	The value to be set as the tape link's **max-partitions** property.
`connectivity`	Integer	Optional	The value to be set as the tape link's **connectivity** property.
`email-to-addresses`	Array of String	Optional	A set of one or more email addresses for the people that are to be notified through email of any new tape link resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent. Default value: **null**. No email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of any new tape link resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the modified tape link resources that require fulfillment. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. An email without a special text insert will be sent.

Description

This operation updates a tape link's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission, or if the **tape-library-uri** field is present and does not reference a Tape Library object to which the API user has object-access permission.

If the CPC on which this tape link resource exists is not active, or if the **connectivity** field value is less than the number of adapter ports in the **adapter-port-uris** property, or if the **max-partitions** field value is less than the number of partitions to which the tape link is currently attached, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This may occur because the parent CPC is already associated with a tape link with the specified name, or because the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or because any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, the tape link's properties are updated to their corresponding request body content's field's values. Optional fields may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified. If the update changes the

value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

If the modified tape link's **fulfillment-state** is **"pending"** and the **email-to-addresses** field is present and not **null** in the request body, an email containing information about the modified tape link resources that require fulfillment or are now unused is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not null, its contents will be inserted into the email body. If an error occurs when sending the email, a 409 (Conflict) status code is returned. This could be because the Console is not configured to support emails. A failure to send the email does not rollback the modification of the tape link. An API client should assume that the tape link was modified even though the request failed with a 409 (Conflict) status code and 491 reason code. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example due to an unknown email address. If a send failure occurs, emails can be resent using the `Resend Request` operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape link-id}*.
- Object-access permission to the tape library referenced by the **tape-library-uri** field, if present in the request body.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 315. Modify Tape Link Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	8	A tape link with the **name** specified in the request body is already associated with the CPC identified by the tape link's **cpc-uri** property.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape link-id}* does not exist on the Console or the API user does not have object-access permission for it.
	2	The tape library identified by **tape-library-uri** does not exist on the Console or the API user does not have object-access permission for it.

Table 315. Modify Tape Link Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The tape link object with the **object-id** *{tape-link-id}* was busy and the request timed out.
	8	The value of the **connectivity** field is less than the number of adapter ports currently referenced in the **adapter-port-uris** property for this tape link.
	475	The **max-partitions** field value is less than the number of partitions to which this tape link is currently attached.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the tape link will have been modified.
	493	The **email-to-addresses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-links/268047d0-39d9-11eb-b329-fa163e874820/operations/modify HTTP/1.1
 x-api-session: 3b6lvzg8t1d3s2wk9kk6rw97qf8vplb6bqwurcrhdseqika5h9
 content-type: application/json
 content-length: 68
 {
    "description":"A modified example tape link",
    "max-partitions":2
 }
```

Figure 360. Modify Tape Link Properties: Request

```
204 No Content
 server: Hardware management console API web server / 2.0
 cache-control: no-cache
 date: Wed, 09 Dec 2020 04:51:30 GMT

<No response body>
```

Figure 361. Modify Tape Link Properties: Response

Delete Tape Link

The `Delete Tape Link` operation deletes a tape link.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/delete
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Optional	A set of one or more email addresses for the people that are to be notified through email of any new tape link resources that are no longer needed. These email addresses will appear in the "to:" address list in the email that is sent. Default value: **null**. No email will be sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of any new tape link resources that are no longer needed. These email addresses will appear in the "cc:" address list in the email that is sent. The **email-cc-addresses** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. No one will be copied on the email.
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the modified tape link resources that are no longer needed. The text can include HTML formatting tags. The **email-insert** field must be **null** when the **email-to-addresses** field is **null**. Default value: **null**. An email without a special text insert will be sent.

Description

This operation deletes a tape link. The tape link must be detached from all active partitions before it can be deleted. An Inventory Change notification for the deleted tape link is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a tape link object to which the API user has object-access permission.

If the **email-insert** or **email-cc-addresses** fields are present in the request body without the **email-to-addresses** field, or if any address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address, a 400 (Bad Request) status code is returned. If the tape link is still attached to any partition, or if the CPC on which this tape link resource exists is not active, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails.

If the request body contents are valid, the identified tape link is deleted from the CPC. If the **email-to-addresses** field is present and not null in the request body, an email containing information about the tape link resources that may now be recovered is sent to the email addresses specified in the **email-to-addresses** and **email-cc-addresses** fields in the request body. If the **email-insert** field is present and not null, its contents will be inserted into the email body. A failure to send the email does not rollback the deletion of the tape link. Note that a successful completion does not imply that the emails were delivered. An API client should assume that the tape link was deleted even though the request failed with a 409 (Conflict) status code and 491 reason code. Errors could be encountered at an email server after the request completes, for example due to an unknown email address.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 316. Delete Tape Link: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	451	The **email-cc-addresses** or **email-insert** field is present in the request body without the **email-to-addresses** field.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The tape link object with the **object-id** *{tape-link-id}* was busy and the request timed out.
	481	The tape link identified by *{tape-link-id}* is still attached to at least one active partition.
	491	An error occurred when sending the email. This failure applies only to the sending of the email. If this reason code is returned, the tape link will have been deleted.
	493	The **email-to-addresses** field is present in the request body, but the targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-links/268047d0-39d9-11eb-b329-fa163e874820/operations/delete HTTP/1.1
 x-api-session: 3iw5ijchght27jfm4m53vavbyg1958bjih06ihhu9cfepohd1c
 content-type: application/json
```

Figure 362. Delete Tape Link: Request

```
204 No Content
 server: Hardware management console API web server / 2.0
 cache-control: no-cache
 date: Wed, 09 Dec 2020 04:53:41 GMT

<No response body>
```

Figure 363. Delete Tape Link: Response

Add Adapter Ports

The Add `Adapter Ports` operation adds a list of storage adapter ports to a tape link.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/add-adapter-ports
```

In this request, the URI variable *{tape-link-id}* is the object ID of the tape link to which the adapter ports are to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uris`	Array of String/URI	Required	A list of the adapter ports that are to be added to the tape link's adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port.

Description

This operation adds a list of storage adapter ports to a tape link's adapter ports list. These adapter ports provide additional bandwidth to the target tape library. The adapter ports should have connectivity to the Storage Area Network (SAN). A change notification for the tape link's **adapter-port-uris** property is emitted asynchronously to this operation.

The number of adapter ports associated with a tape link cannot exceed the value of its **connectivity** property. It might be necessary to first modify the tape link to increase that property value before adding new adapter ports.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a tape link to which the API user has object-access permission, or if any

member of the **adapter-port-uris** list does not identify a port in a storage adapter to which the API user has object-access permission.

If any adapter port in the **adapter-port-uris** list is currently a member of the tape link's adapter ports list, or if any adapter port in the **adapter-port-uris** list references a storage adapter port that does not reside in the target tape link's CPC, or if any adapter port in the **adapter-port-uris** list references an adapter that is not an FCP storage adapter, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are added to the tape link's adapter ports list.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Object-access permission to each adapter containing the ports identified in the **adapter-port-uris** array.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 317. Add Adapter Ports: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission for it.
	2	An adapter containing a port referenced by a URI in the **adapter-port-uris** field does not exist on the HMC or the API user does not have object-access permission for it.

Table 317. Add Adapter Ports: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The tape link object with the **object-id** *{tape-link-id}* was busy and the request timed out.
	8	The modification would put the tape link object with the **object-id** *{tape-link-id}* in a state where the number of adapter ports in its **adapter-port-uris** property would exceed the maximum number of connections as specified in its **connectivity** property.
	441	A storage adapter port in the **adapter-port-uris** list resides in a different CPC than the targeted tape link.
	478	A storage adapter port in the **adapter-port-uris** list is already a current member of the **adapter-port-uris** list of the tape link's object with the **object-id** *{tape-link-id}*.
	483	The adapter that contains the port referenced by an element in the **adapter-port-uris** field has a **type** value other than **"fcp"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903/operations/add-adapter-ports HTTP/1.1
x-api-session: 1wlfu8y6prhs82nd9hwb2acazs3q410obcqxarv0nrita8vaed
content-type: application/json
content-length: 93
{
   "adapter-port-uris":[
      "/api/adapters/8e8b7b10-4e6f-11eb-b5a0-fa163ed4d903/storage-ports/0"
   ]
}
```

Figure 364. Add Adapter Ports: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 14:48:07 GMT

<No response body>
```

Figure 365. Add Adapter Ports: Response

Remove Adapter Ports

The `Remove Adapter Ports` operation removes a list of storage adapter ports from a tape link.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/remove-adapter-ports
```

In this request, the URI variable *{tape-link-id}* is the object ID of the tape link from which the adapter ports are to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`adapter-port-uris`	Array of String/ URI	Required	A list of the adapter ports that are to be removed from the tape link's adapter ports list. Each element in this array is an instance of the canonical URI path of a storage adapter port.

Description

This operation removes a list of storage adapter ports from a tape link’s adapter ports list. This removes bandwidth to the target tape library. A change notification for the tape link's **adapter-port-uris** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a tape link to which the API user has object-access permission.

If any adapter port in the **adapter-port-uris** list is not currently a member of the tape link's adapter ports list, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, the adapter port URIs in the **adapter-port-uris** list are removed from the tape link's adapter ports list.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 318. Remove Adapter Ports: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.

Table 318. Remove Adapter Ports: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The tape link object with the **object-id** *{tape-link-id}* was busy and the request timed out.
	479	A storage adapter port in the **adapter-port-uris** list is not a current member of the **adapter-port-uris** list of the tape link's object with the **object-id** *{tape-link-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903/operations/remove-adapter-ports
  HTTP/1.1
x-api-session: 2ownc0wu8b95atkkpksvcgif0h5wxmfdmnf8mfh93qdnnkio5f
content-type: application/json
content-length: 93
{
   "adapter-port-uris":[
      "/api/adapters/8e8b7b10-4e6f-11eb-b5a0-fa163ed4d903/storage-ports/0"
   ]
}
```

Figure 366. Remove Adapter Ports: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 14:49:27 GMT

<No response body>
```

Figure 367. Remove Adapter Ports: Response

Replace Adapter Port

The `Replace Adapter Port` operation replaces an existing adapter port in a tape link's list of storage adapter ports with a different adapter port.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/replace-adapter-port
```

In this request, the URI variable *{tape-link-id}* is the object ID of the tape link for which the adapter port is to be replaced.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`old-adapter-port-uri`	String/URI	Required	The canonical URI path of a storage adapter port in the target tape link's **adapter-port-uris** list that is to be replaced.
`new-adapter-port-uri`	String/URI	Required	The canonical URI path of a storage adapter port that is to replace the one referenced by **old-adapter-port-uri**.

Description

This operation replaces one of the storage adapter ports in a tape link's adapter ports list. This operation differs from issuing separate remove and add adapter ports operations in that it guarantees that the device numbers are transferred to the new configuration. Specifically, each partition to which the target tape link is attached will be associated to a virtual tape resource with the new backing adapter port, but with the same device number. A change notification for the tape link's **adapter-port-uris** property is emitted asynchronously to this operation.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a tape link to which the API user has object-access permission, or if **new-adapter-port-uri** does not identify a port in a storage adapter to which the API user has object-access permission.

If the adapter port identified by **old-adapter-port-uri** is not currently a member of the tape link’s adapter ports list, or if the adapter port referenced by **new-adapter-port-uri** is currently a member of the tape link's adapter ports list, or does not reside in the target tape link's CPC, a 409 (Conflict) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the adapter referenced by **new-adapter-port-uri** is not a FICON adapter configured in FCP mode.

If the request body contents are valid, the adapter port identified by **old-adapter-port-uri** is replaced by the one identified by **new-adapter-port-uri** in the tape link's adapter ports list.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Object-access permission to the adapter containing the port identified in **new-adapter-port-uri**.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 319. Replace Adapter Port: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission for it.
	2	The adapter containing the port referenced by the URI in the **new-adapter-port-uri** field does not exist on the HMC or the API user does not have object-access permission for it.
	6	The adapter that contains the port referenced by the **new-adapter-port-uris** field has a **type** value other than **"fcp"**.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	441	The storage adapter port identified by **new-adapter-port-uri** resides in a different CPC than the targeted tape link.
	478	The storage adapter port identified by **new-adapter-port-uri** is already a current member of the **adapter-port-uris** list of the tape link's object with the **object-id** *{tape-link-id}*.
	479	The storage adapter port identified by **old-adapter-port-uri** is not a current member of the **adapter-port-uris** list of the tape link's object with the **object-id** *{tape-link-id}*.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903/operations/replace-adapter-port
  HTTP/1.1
x-api-session: 5bsr4n7zufpqax0jnvsy0h4j4yg5em3uo1pbxqfsr67iw0zfib
content-type: application/json
content-length: 188
{
   "new-adapter-port-uri":"/api/adapters/8e3696e0-4e6f-11eb-b5a0-fa163ed4d903/storage-ports/0",
   "old-adapter-port-uri":"/api/adapters/8e8b7b10-4e6f-11eb-b5a0-fa163ed4d903/storage-ports/0"
}
```

Figure 368. Replace Adapter Port: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 14:52:36 GMT

<No response body>
```

Figure 369. Replace Adapter Port: Response

Resend Request

The `Resend Request` operation requests fulfillment of the tape link resources that require fulfillment. This operation can be invoked on any tape link with fulfillment state of **"pending"**, **"incomplete"**, or **"pending-with-mismatches"**. The request will include the current resources that require fulfillment, and will therefore reflect any subsequent modifications or partial fulfillments that have occurred since the tape link was initially created or modified.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/resend-request
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object for which a fulfillment request should be sent.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`email-to-addresses`	Array of String	Required	A set of one or more email addresses of the people that are to be notified through email of the tape link resources that require fulfillment. These email addresses will appear in the "to:" address list in the email that is sent.
`email-cc-addresses`	Array of String	Optional	A set of zero or more email addresses for the people that are to be copied on the email notification of the tape link resources that require fulfillment. These email addresses will appear in the "cc:" address list in the email that is sent. Default value: **null**. No one will be copied on the email.

Field name	Type	Rqd/Opt	Description
`email-insert`	String	Optional	Text that is to be inserted in the email notification of the tape link resources that require fulfillment. The text can include HTML formatting tags. Default value: **null**. An email without a special text insert will be sent.

Description

This operation notifies through email the new, deleted or modified tape link resources that require fulfillment action by the SAN administrator. The email will reflect the current fulfillment state of the tape link. The content of the email may differ from the email that was sent as a result of a previous `Create Tape Link` or `Modify Tape Link Properties` operation in the following ways:

- If the tape link was modified since the last time it was fulfilled, the email will include the accumulation of all tape link resource changes from the previous modify requests.
- The tape link resources that were part of the previous modify request but are now fulfilled will be omitted from the email.
- Tape link resources that were deleted in a modify request may not be included in the email.

If the API user does not have action/task permission to the **Configure Storage - System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission.

If the current tape link does not contain any new, deleted or modified resources that require fulfillment, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if there is an error sending the email. This could be because the HMC is not configured to send emails. Note that a successful completion does not imply that the emails were delivered. Errors could be encountered at an email server after the request completes, for example, due to an unknown email address.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because an address in the **email-to-addresses** or **email-cc-addresses** fields is not a valid email address.

If the request body contents are valid, an email is created that describes the new or modified tape link resources that require fulfillment, as well as any deleted tape link resources that are no longer needed. If the **email-insert** field is present in the request body, its value is inserted into the body of the email. The email is then sent to the email addresses specified in the **email-to-addresses** field and copied to the email addresses specified in the **email-cc-addresses** field.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 320. Resend Request: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	1	The tape link is not in a valid state to perform the operation (must be in one of the following fulfillment states: **"pending"** (excluding the condition where the tape link is pending because of pending attachment or detachment of partitions), **"incomplete"** (due to one of the reasons listed under **"incomplete-reasons"** of tape link excluding **"adapter-degraded"** and **"adapter-link-error"**) or **"pending-with-mismatches"**.
	491	An error occurred when sending the email.
	493	The targeted console is not configured to support the sending of emails.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/tape-links/03e4e62c-5905-11eb-9244-00106f23d200/operations/resend-request HTTP/1.1
x-api-session: 4v8m2o2idf0t4k7h2bryxatrf0orgvwwcabxntip6ltck6ei1b
content-type: application/json
content-length: 51
{
   "email-to-addresses":[
      "exampleEmail@in.example.com"
   ]
}
```

Figure 370. Resend Request: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 19 Jan 2021 12:35:19 GMT

<No response body>
```

Figure 371. Resend Request: Response

List Virtual Tape Resources of a Tape Link

The `List Virtual Tape Resources of a Tape Link` operation lists the virtual tape resources of the FCP tape link with the given identifier.

HTTP method and URI

```
GET /api/tape-links/{tape-link-id}/virtual-tape-resources
```

In this request, the URI variable *{tape-link-id}* is the **object-id** of the Tape Link object.

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`device-number`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **device-number** property.
`adapter-port-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **adapter-port-uri** property.
`partition-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **partition-uri** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`virtual-tape-resources`	Array of virtual-tape-resource-info objects	Array of virtual-tape-resource-info objects, described in the next table. The returned array may be empty.

Each nested virtual-tape-resource-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The canonical URI path (**element-uri**) of the Virtual Tape Resource element object.
`name`	String	The **name** property of the Virtual Tape Resource element object.
`device-number`	String	The **device-number** property of the Virtual Tape Resource element object.
`adapter-port-uri`	String/ URI	The **adapter-port-uri** property of the Virtual Tape Resource element object.
`partition-uri`	String/ URI	The **partition-uri** property of the Virtual Tape Resource element object.

Description

This operation lists the virtual tape resources that are owned by the identified tape link. The element URI, name, device number and associated adapter port and partition are provided for each.

If the object ID *{tape-link-id}* does not identify a tape link object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

If the **name** or **device-number** query parameter is specified, the returned list is limited to those virtual tape resources that have a **name** or **device-number** property matching the specified filter pattern. If the **name** or **device-number** parameter is omitted, the filtering on the omitted property name is not done.

If the **adapter-port-uri** or **partition-uri** query parameter is specified, the returned list is limited to those virtual tape resources that have a matching **adapter-port-uri** or **partition-uri** property. If the **adapter-port-uri** or **partition-uri** parameter is omitted, the filtering on the omitted property name is not done.

If no virtual tape resources are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 707.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 321. List Virtual Tape Resources of a Tape Link: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	The tape link with the object ID *{tape-link-id}* does not exist on the Console, or the API user does not have object-access permission for it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9/virtual-tape-resources HTTP/1.1
x-api-session: 4176um534e2nvp7vjun34lr2cf9bxtk4lvh4t3un9i3le31347
```

Figure 372. List Virtual Tape Resources of a Tape Link: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 04 Jan 2021 12:15:21 GMT
content-type: application/json;charset=UTF-8
content-length: 379
{
   "virtual-tape-resources":[
      {
         "adapter-port-uri":"/api/adapters/ac5207aa-4fc4-11e9-b8fd-00106f0d81c9/
           storage-ports/0",
         "device-number":"000b",
         "element-uri":"/api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9/
           virtual-tape-resources/0f576e86-3a23-11eb-9352-00106f0d81c9",
         "name":"vhba_TL_Connect2TapeTL1PAth1",
         "partition-uri":"/api/partitions/1118e03e-39e1-11eb-8ca5-00106f0d81c9"
      }
   ]
}
```

Figure 373. List Virtual Tape Resources of a Tape Link: Response

Get Virtual Tape Resource Properties

The `Get Virtual Tape Resource Properties` operation retrieves the properties of a single Virtual Tape Resource element object.

HTTP method and URI

```
GET /api/tape-links/{tape-link-id}/virtual-tape-resources/{virtual-tape-resource-id}
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object and the URI variable *{virtual-tape-resource-id}* is the element ID of the Virtual Tape Resource element object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Virtual Tape Resource element object as defined in the "Virtual Tape Resource element object" on page 681. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

Returns the current values of the properties for the Virtual Tape Resource element object as defined in "Virtual Tape Resource element object" on page 681.

A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission, or if the element ID *{virtual-tape-resource-id}* does not identify a virtual tape resource in the tape link.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 709.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 322. Get Virtual Tape Resource Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	A tape link with **object-id** *{tape-link-id}* does not exist on the Console, or the API user does not have object-access permission to it.
	5	A virtual tape resource with **element-id** *{virtual-tape-resource-id}* does not exist in the tape link on the Console.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9/virtual-tape-resources/
  0f2b57f6-3a23-11eb-9352-00106f0d81c9 HTTP/1.1
x-api-session: 5ugs7k2mdg9x4thu6e8y5ih1uk9xxa52yp7ud1rpvq3fl3kn30
```

Figure 374. Get Virtual Tape Resource Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 04 Jan 2021 13:00:14 GMT
content-type: application/json;charset=UTF-8
content-length: 630
{
   "adapter-port-uri":"/api/adapters/ab61e64e-4fc4-11e9-b8fd-00106f0d81c9/storage-ports/0",
   "class":"virtual-tape-resource",
   "degraded-reasons":[],
   "description":"",
   "device-number":"000a",
   "element-id":"0f2b57f6-3a23-11eb-9352-00106f0d81c9",
   "element-uri":"/api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9/virtual-tape-resources/
     0f2b57f6-3a23-11eb-9352-00106f0d81c9",
   "name":"vhba_TL_Connect2TapeTL1PAth0",
   "parent":"/api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9",
   "partition-uri":"/api/partitions/1118e03e-39e1-11eb-8ca5-00106f0d81c9",
   "world-wide-port-name-info":{
      "status":"validated",
      "world-wide-port-name":"c05076ffe800000f"
   }
}
```

Figure 375. Get Virtual Tape Resource Properties: Response

Update Virtual Tape Resource Properties

The `Update Virtual Tape Resource Properties` operation updates one or more of the writable properties of a virtual tape resource.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/virtual-tape-resources/{virtual-tape-resource-id}
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object and the URI variable *{virtual-tape-resource-id}* is the element ID of the virtual tape resource element.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined in the "Virtual Tape Resource element object" on page 681. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates a virtual tape resource's properties with the values specified.

If the API user does not have action/task permission to the **Configure Storage – System Programmer** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission, or if the element ID *{virtual-tape-resource-id}* does not identify a Virtual Tape Resource element object in the tape link. In addition to object-access to the parent tape link, updates to the **device-number** property also require object-access permission to the partition that is associated with the target virtual tape resource. If this property appears in the request body and the user does not have object-access to the partition identified in the **partition-uri** property, a 403 (Forbidden) status code is returned. If the CPC on which this virtual tape resource exists is not active, a 409 (Conflict) status code is returned.

If the request body contents are valid, the virtual tape resource's properties are updated to their corresponding request body content's field's values. All fields are optional and may be excluded from the request body; if a field is not found in the request body, its property's value will not be modified.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Object-access permission to the partition referenced by the virtual tape resource's **partition-uri** property. This requirement only applies when updating the **device-number** property.
- Action/task permission to the **Configure Storage – System Programmer** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 323. Update Virtual Tape Resource Properties: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A virtual tape resource with the **name** specified in the request body already exists within the parent tape link.
	452	The new value supplied in the **device-number** field conflicts with an existing device number for another device attached to the partition associated with this virtual tape resource.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** task.
	450	The **device-number** fields are present in the request body and the API user does not have object-access permission to the partition referenced by the current value of the virtual tape resource's **partition-uri** property
404 (Not Found)	1	A tape link with **object-id** *{tape-link-id}* does not exist on the Console, or the API user does not have object-access permission to it.
	5	A virtual tape resource with **element-id** *{virtual-tape-resource-id}* does not exist in the tape link on the Console.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**).
	2	The parent tape link with **object-id** *{tape-link-id}* or the partition identified by this virtual tape resource's **partition-uri** property value was busy and the request timed out.
	6	The **status** property value for the partition identified by this virtual tape resource's **partition-uri** property is not valid to perform the operation (must be in one of the following states: **"active"**, **"degraded"**, **"paused"**, **"reservation-error"**, **"stopped"**, or **"terminated"**).
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/tape-links/4e1293ae-3a22-11eb-9352-00106f0d81c9/virtual-tape-resources/
  0f2b57f6-3a23-11eb-9352-00106f0d81c9 HTTP/1.1
x-api-session: 4oko6xm0vyr1xo864ic5eis6fvvznqcx55x10rgkqsx6yb1l1v
content-type: application/json
content-length: 51
{
   "description":"A modified virtual tape resource"
}
```

Figure 376. Update Virtual Tape Resource Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 04 Jan 2021 13:34:14 GMT

<No response body>
```

Figure 377. Update Virtual Tape Resource Properties: Response

Get Partitions for a Tape Link

The `Get Partitions for a Tape Link` operation lists the partitions to which the tape link with the given identifier is attached.

HTTP method and URI

```
GET /api/tape-links/{tape-link-id}/operations/get-partitions
```

In this request, the URI variable *{tape-link-id}* is the **object-id** of the Tape Link object.

Query parameters

Name	Type	Req/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned Partition objects to those that have a matching **name** property
`status`	String Enum	Optional	Optional filter string to limit returned Partition objects to those that have a matching **status** property. Value must be a valid partition **status** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`partitions`	Array of partition-info objects	Array of partition-info objects, described in the next table. The returned array may be empty.

Each nested partition-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The canonical URI path (**object-uri**) of the Partition object. This property will be null when the current user has no object access permission to the partition.
`name`	String	The **name** property of the Partition object.
`status`	String Enum	The **status** property of the Partition object.

Description

This operation lists the partitions to which the identified tape link is attached. The object URI, name and status are provided for each.

If the object ID *{tape-link-id}* does not identify a tape link object to which the API user has object-access permission, a 404 (Not Found) status code is returned.

If the **name** query parameter is specified, the returned list is limited to those partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid value for the **status** property according to the Partition object data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partitions that have the specified **status** value. If the **status** parameter is omitted, this filtering is not done.

The operation lists all partitions attached to the tape link including partitions the current user may not have object access permission to. For partitions without object access permission, the **object-uri** property will be null. If no partitions are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 713.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 324. Get Partitions for a Tape Link: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.
404 (Not Found)	1	A tape link with **object-id** *{tape-link-id}* does not exist on the Console or the API user does not have object-access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links/43750b00-4f64-11eb-b10a-fa163ed4d903/operations/get-partitions HTTP/1.1
x-api-session: 3trdm1tc3mz0g9z8u3t2vkj0v6t9cak6649nue3igkullhrdn4
```

Figure 378. Get Partitions for a Tape Link: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 05 Jan 2021 15:01:16 GMT
content-type: application/json;charset=UTF-8
content-length: 123
{
   "partitions":[
      {
         "name":"Database",
         "object-uri":"/api/partitions/f1ff01b6-4f65-11eb-b6ac-fa163ed4d903",
         "status":"stopped"
      }
   ]
}
```

Figure 379. Get Partitions for a Tape Link: Response

Get Tape Link Histories

The `Get Tape Link Histories` operation returns a chronological list of actions performed on the tape links known to the target Console. The response can be filtered according to query parameters, if specified.

HTTP method and URI

```
GET /api/console/operations/get-tape-link-histories
```

Query parameters:

Name	Type	Rqd/ Opt	Description
`begin-time`	Timestamp	Optional	A timestamp to filter the entries returned in the **tape-link-actions** list. Actions with **action-time** values earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp to filter the entries returned in the **tape-link-actions** list. Actions with **action-time** values later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`cpc-uri`	String/ URI	Optional	The canonical URI path of a CPC object to limit the returned entries to those that have a matching **cpc-uri** field. If not specified, then no such filtering is performed.
`tape-link-uri`	String/ URI	Optional	The canonical URI path of a Tape Link object to limit the returned entries to those that have a matching **tape-link-uri** field. If not specified, then no such filtering is performed.

Name	Type	Rqd/ Opt	Description
`user-name`	String	Optional	Filter pattern (regular expression) to limit the entries returned in the **tape-link-actions** list to those that have a matching **user-name** field. If not specified, then no such filtering is performed.
`tape-link-name`	String	Optional	Filter pattern (regular expression) to limit the entries returned in the **tape-link-actions** list to those that have a matching **tape-link-name** field. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`tape-link-histories`	Array of tape-link-history-info objects	The list of information about the history of actions performed on each tape link. Each element in the array is an instance of tape-link-history-info object, described in the next table. The returned array may be empty.

Each tape-link-history-info object contains the following fields:

Table 325. tape-link-history-info nested object

Name	Type	Description of specialization
`cpc-uri`	String/URI	The canonical URI path of the CPC object with which the tape link is associated.
`tape-link-uri`	String/URI	The canonical URI path of the Tape Link object with which the tape link history is associated.
`tape-link-actions`	Array of tape-link-action-info objects	The list of information about the actions performed on the tape link. Each element in the array is an instance of a tape-link-action-info object, described in the next table.
`tape-link-name`	String (1-64)	The display name specified for the tape link with which the tape link history is associated. Refer to the **name** property of the Tape Link object defined in Table 309 on page 677.

Each tape-link-action-info object contains the following fields:

Table 326. tape-link-action-info nested object

Name	Type	Description
`user-name`	String (4-320)	The name of the user who performed the action on the tape link. Refer to the **name** property of the User object, defined in Table 456 on page 893 The value is **"no user"** if the action was performed by the system.
`action-time`	Timestamp	The time when the action was performed on the tape link.

Table 326. tape-link-action-info nested object (continued)		
Name	**Type**	**Description**
`action`	String Enum	The action that was performed on the tape link. Values: • **"creation-requested"** - The tape link was created. • **"modification-requested"** - One or more properties of the tape link were modified. Some properties are changed immediately and some are pending fulfillment which is indicated by the field **tape-link-fulfillment-state**. • **"deletion-requested"** - The tape link was deleted. • **"request-fulfilled"** - The tape link request was fulfilled. • **"attached-to-partition"** - The tape link was attached to a partition. • **"detached-from-partition"** - The tape link was detached from a partition. • **"send-request-failed"** - A send request operation failed during the creation, modification or deletion of the tape link, or of a subsequent resend. • **"backing-adapter-changed"** - The backing adapter of a virtual tape resource belonging to the tape link was changed. • **"partition-name-changed"** - The name of a Partition to which this tape link is attached was changed. • **"device-number-changed"** - A device number of a virtual tape resource belonging to the tape link was changed. • **"adapter-degraded"** - One of the required adapters of the tape link is degraded. • **"adapter-degradation-resolved"** - Degradation was resolved for one of the required adapters in the tape link. • **"adapter-mismatches-discovered"** - Adapters that were not requested for the tape link were discovered. • **"adapter-mismatch-resolved"** - Adapter mismatches were resolved for tape link. • **"adapter-mismatch-partially-resolved"** - Adapter mismatches were partially resolved for tape link. • **"zoning-error"** – A zoning error was found for the tape link. • **"requested-library-not-discovered"** – The library that was requested for the tape link was not discovered. • **"multiple-libraries-discovered"** - More than one library was discovered for the requested tape link. • **"unknown-library-discovered"** - Drives were sensed that could not be associated with a library. • **"different-library-discovered"** - A tape library was discovered that was not requested for the tape link. • **"no-library-discovered"** - No libraries were discovered for the tape link.

Table 326. tape-link-action-info nested object (continued)		
Name	**Type**	**Description**
`tape-link-fulfillment-state`	String Enum	The fulfillment state of the tape link, after the action was performed. Values: • Any value of the **fulfillment-state** property as defined in the Tape Link Data model in Table 310 on page 678. • **"deleted"** - The tape link has been deleted.
`tape-link-configuration`	tape-link-configuration object	A nested object that provides additional information about the configuration of the tape link that was changed as part of this action. The value is a single instance of a tape-link-configuration object, defined in Table 327 on page 718. The fields in this object are present only when the corresponding property is modified in this action, unless mentioned otherwise explicitly in the description. This field is only present when the value of the **action** property is one of the following: **"creation-requested"**, **"modification-requested"**, **"deletion-requested"**, **"attached-to-partition"**, **"detached-from-partition"**, **"backing-adapter-changed"**, **"device-number-changed"**, **"adapter-mismatch-resolved"**, or **"adapter-mismatch-partially-resolved"**. When the action is **"creation-requested"**, the fields in this object will be populated similar to the **"modification-requested"** action, considering all values of the create request as new or added, When the action is **"deletion-requested"**, the fields in this object will be populated similar to the **"modification-requested"** action , considering all values of the delete request as old or deleted.
`partition-uri`	String/URI	The canonical URI path of the partition which this tape link is attached to, detached from, or name changed when attached with this tape link This field is only present if the **action** property is either **"attached-to-partition"**, **"detached-from-partition"**, or **"partition-name-changed"**.
`old-partition-name`	String (1-64)	The old name of the partition to which the tape link is attached. This field is only present if the **action** property is **"partition-name-changed"**.
`new-partition-name`	String (1-64)	The new name of the partition to which the tape link is attached. This field is only present if the **action** property is **"partition-name-changed"**.

Each tape-link-configuration object contains the following fields:

Table 327. tape-link-configuration nested object		
Name	**Type**	**Description**
`old-name`	String (1-64)	The old value for the **name** property of the tape link.
`new-name`	String (1-64)	The new value for the **name** property of the tape link.

Table 327. tape-link-configuration nested object (continued)		
Name	**Type**	**Description**
`old-description`	String (0-200)	The old value for the **description** property of the tape link.
`new-description`	String (0-200)	The new value for the **description** property of the tape link.
`old-connectivity`	Integer	The old value for the **connectivity** property of the tape link.
`new-connectivity`	Integer	The new value for the **connectivity** property of the tape link.
`old-max-partitions`	Integer	The old value for the **max-partitions** property of the tape link.
`new-max-partitions`	Integer	The new value for the **max-partitions** property of the tape link.
`tape-library-uri`	String/ URI	The canonical URI path of the tape library object linked by this tape link object. This value will be **null**, if a tape library was not specified when creating the tape link. This field is only present if the **action** property is **"create-requested"**.
`world-wide-port-names-added`	Array of world-wide-port-name-info objects	The list of information about the worldwide port names (WWPNs) that have been newly allocated to this tape link. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 240 on page 544. This field is only present when the action resulted in new worldwide port names being added for this tape link.
`world-wide-port-names-deleted`	Array of world-wide-port-name-info objects.	The list of information about the worldwide port names (WWPNs) that have been deleted from this tape link. Each element in this array is an instance of a world-wide-port-name-info nested object defined in Table 240 on page 544. This field is only present when the action resulted in worldwide port names being deleted from this tape link.
`virtual-tape-resources-added`	Array of virtual-tape-resource-info objects	The list of information about the virtual tape resources of this tape link that were added. Each element in this array is an instance of a virtual-tape-resource-info object, defined in Table 328 on page 721. This field is only present when the **action** property is either **"attached-to-partition"** or **"modification-requested"** and the action resulted in virtual tape resources of this tape link being added.

Table 327. tape-link-configuration nested object (continued)		
Name	**Type**	**Description**
`virtual-tape-resources-modified`	Array of virtual-tape-resource-info objects	The list of information about the virtual tape resources of this tape link that were modified. Each element in this array is an instance of a virtual-tape-resource-info object, defined in Table 328 on page 721. The fields in this object are present only when the virtual tape resource's corresponding property is modified in this action, unless explicitly mentioned otherwise in the description. This field is only present when the **action** property is either **"device-number-changed"**, **"backing-adapter-changed"** or **"modification-requested"** and the action resulted in virtual tape resources of this tape link being modified.
`virtual-tape-resources-deleted`	Array of virtual-tape-resource-info objects	The list of information about the virtual tape resources of this tape link that were deleted. Each element in this array is an instance of a virtual-tape-resource-info object, defined in Table 328 on page 721. This field is only present when the **action** property is either **"detached-from-partition"** or **"modification-requested"** and the action resulted in virtual tape resources of this tape link being deleted.
`adapters-added`	Array of String/ URI	The list of requested storage adapter ports for this tape link. Each element in this array is the canonical URI path of an added Adapter Port element object. This field is only present when the **action** property is **"modification-requested"** and the action resulted in one or more adapters being added to the tape link.
`adapters-removed`	Array of String/ URI	The list of storage adapter ports for this tape link that are removed. Each element in this array is the canonical URI path of a removed Adapter Port element object. This field is only present when the **action** property is **"modification-requested"** and the action resulted in one or more adapters being removed from the tape link.
`partitions-added`	Array of String/ URI	The list of partitions added for the tape link. Each element in this array is the canonical URI path of an added Partition object. This field is only present when the **action** property is either **“modification-requested"** or **"creation-requested"** and the action resulted in one or more partitions being added to the tape link.
`partitions-removed`	Array of String/ URI	The list of partitions removed from the tape link. Each element in this array is the canonical URI path of a removed Partition object. This field is only present when the **action** property is **"modification-requested"** and the action resulted in one or more partitions being removed from the tape link.

Each virtual-tape-resource-info object contains the following fields:

Table 328. virtual-tape-resource-info nested object		
Name	**Type**	**Description**
`element-uri`	String/URI	The canonical URI path for the Virtual Tape Resource element object that was modified in this action.
`old-name`	String (1-64)	The old value for the **name** property of the virtual tape resource.
`new-name`	String (1-64)	The new value for the **name** property of the virtual tape resource.
`old-description`	String (0-200)	The old value for the **description** property of the virtual tape resource.
`new-description`	String (0-200)	The new value for the **description** property of the virtual tape resource.
`partition-uri`	String/URI	The value of the **partition-uri** property of the Virtual Tape Resource element object at the time of this action. This field is always present.
`old-device-number`	String (4)	The old value for the **device-number** property of the virtual tape resource.
`new-device-number`	String (4)	The new value for the **device-number** property of the virtual tape resource.
`old-adapter-port-uri`	String/URI	The old value for the **adapter-port-uri** property of the virtual tape resource. The value will be **null** when the **adapter-port-uri** is not previously specified for the virtual tape resource.
`new-adapter-port-uri`	String/URI	The new value for the **adapter-port-uri** property of the virtual tape resource. The value will be **null** when the **adapter-port-uri** is not specified for the virtual tape resource.
`world-wide-port-name`	String (16)	The value of the **world-wide-port-name** property of the Virtual Tape Resource element object at the time of this action.

Description

This operation returns information about the history of actions performed for all the tape links known to this console. It contains a chronological list of actions performed on tape links. Information about tape links to which the API user does not have object-access permission, are not included in the response. The history entry for each tape link contains an array of tape-link-action-info objects representing the actions performed on the tape link.

The order in which the tape-link-history-info objects are returned is not guaranteed and can change for every request.

A tape link's history entries are not immediately deleted when the tape link object is deleted. The history entries are preserved for 30 days from the deletion of the tape link object, after which they are deleted automatically. This allows API clients to query minimal configuration information for the tape link even after they receive notification that the tape link has been deleted. API clients can use this configuration information to undo any configuration done earlier on the SAN in order to fulfill this tape link.

The entries can be limited by specifying explicit filtering criteria on the request. Filtering can occur on two different levels. The **tape-link-uri**, **tape-link-name** and **cpc-uri** query parameters filter the tape links for which history information is returned. The **begin-time**, **end-time** and **user-name** query parameters filter the actions that are returned for each tape link

If the **begin-time** query parameter is specified, then any actions earlier than that time are omitted. If the **end-time** query parameter is specified, then any actions later than that time are omitted.

If the **tape-link-uri** query parameter is specified for the request, the returned list is limited to the tape link history entry that has a matching **tape-link-uri** value.

If the **cpc-uri** query parameter is specified for the request, the returned list is limited to the tape link history entries that have a matching **cpc-uri** value.

If the **user-name** query parameter is specified for the request, only actions with a **user-name** property matching the specified filter pattern are returned.

If the **tape-link-name** parameter is specified for the request, the returned list is limited to the tape link history entries that have a **tape-link-name** property matching the specified filter pattern.

If no tape link history entries are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Tape Link object designated by the **tape-link-uri** property of each tape-link-history-info object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 716.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 329. Get Tape Link Histories: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter contains an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-tape-link-histories HTTP/1.1
x-api-session: 2mdml78jyprziog98oagcqsy7uis3cnfcgkf4hglv7voynxqfz
```

Figure 380. Get Tape Link Histories: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 25 Jan 2021 13:13:00 GMT
content-type: application/json;charset=UTF-8
content-length: 4129
{
   "tape-link-histories":[
      {
         "cpc-uri":"/api/cpcs/ea21f012-7597-3f35-a091-6d94577d5f50",
         "tape-link-actions":[
            {
               "action":"creation-requested",
               "action-time":1611579981424,
               "tape-link-configuration":{
                  "new-connectivity":2,
                  "new-description":"",
                  "new-max-partitions":1,
                  "new-name":"first-tape",
                  "tape-library-uri":null,
                  "world-wide-port-names-added":[
                     {
                        "status":"not-validated",
                        "world-wide-port-name":"a1b2c3d4e5f60002"
                     },
                     {
                        "status":"not-validated",
                        "world-wide-port-name":"a1b2c3d4e5f60003"
                     }
                  ]
               },
               "tape-link-fulfillment-state":"pending",
               "user-name":"SYSPROG"
            },
            {
               "action":"modification-requested",
               "action-time":1611580074356,
               "tape-link-configuration":{
                  "partitions-added":[
                     "/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5"
                  ]
               },
               "tape-link-fulfillment-state":"pending",
               "user-name":"SYSPROG"
            },
            {
               "action":"attached-to-partition",
               "action-time":1611580076205,
               "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
               "tape-link-configuration":{
                  "virtual-tape-resources-added":[
                     {
                        "element-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5/
                          virtual-tape-resources/56c45314-5f0e-11eb-9e13-fa163edda9b5",
                        "new-adapter-port-uri":null,
                        "new-description":"",
                        "new-device-number":"0002",
                        "new-name":"vhba_TL_first-tape1",
                        "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
                        "world-wide-port-name":"a1b2c3d4e5f60003"
                     },
```

Figure 381. Get Tape Link Histories: Response (Part 1)

```
                    {
                        "element-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5/
                          virtual-tape-resources/56430ce6-5f0e-11eb-9e13-fa163edda9b5",
                        "new-adapter-port-uri":null,
                        "new-description":"",
                        "new-device-number":"0001",
                        "new-name":"vhba_TL_first-tape0",
                        "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
                        "world-wide-port-name":"a1b2c3d4e5f60002"
                    }
                ]
            },
            "tape-link-fulfillment-state":"pending",
            "user-name":"SYSPROG"
        },
        {
            "action":"detached-from-partition",
            "action-time":1611580111959,
            "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
            "tape-link-configuration":{
                "virtual-tape-resources-deleted":[
                    {
                        "element-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5/
                          virtual-tape-resources/56430ce6-5f0e-11eb-9e13-fa163edda9b5",
                        "old-adapter-port-uri":null,
                        "old-description":"",
                        "old-device-number":"0001",
                        "old-name":"vhba_TL_first-tape0",
                        "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
                        "world-wide-port-name":"a1b2c3d4e5f60002"
                    },
                    {
                        "element-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5/
                          virtual-tape-resources/56c45314-5f0e-11eb-9e13-fa163edda9b5",
                        "old-adapter-port-uri":null,
                        "old-description":"",
                        "old-device-number":"0002",
                        "old-name":"vhba_TL_first-tape1",
                        "partition-uri":"/api/partitions/35a281ce-5f0e-11eb-856c-fa163edda9b5",
                        "world-wide-port-name":"a1b2c3d4e5f60003"
                    }
                ]
            },
            "tape-link-fulfillment-state":"pending",
            "user-name":"SYSPROG"
        },
        {
            "action":"deletion-requested",
            "action-time":1611580112729,
            "tape-link-configuration":{
                "old-connectivity":2,
                "old-description":"",
                "old-max-partitions":1,
                "old-name":"first-tape",
                "world-wide-port-names-deleted":[
```


Figure 382. Get Tape Link Histories: Response (Part 2)

```
                    {
                       "status":"not-validated",
                       "world-wide-port-name":"a1b2c3d4e5f60003"
                    },
                    {
                       "status":"not-validated",
                       "world-wide-port-name":"a1b2c3d4e5f60002"
                    }
                 ]
              },
              "tape-link-fulfillment-state":"deleted",
              "user-name":"SYSPROG"
           }
        ],
        "tape-link-name":"first-tape",
        "tape-link-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5"
     },
     {
        "cpc-uri":"/api/cpcs/ea21f012-7597-3f35-a091-6d94577d5f50",
        "tape-link-actions":[
           {
              "action":"creation-requested",
              "action-time":1611580130447,
              "tape-link-configuration":{
                 "new-connectivity":2,
                 "new-description":"",
                 "new-max-partitions":1,
                 "new-name":"second-tape",
                 "tape-library-uri":null,
                 "world-wide-port-names-added":[
                    {
                       "status":"not-validated",
                       "world-wide-port-name":"a1b2c3d4e5f60006"
                    },
                    {
                       "status":"not-validated",
                       "world-wide-port-name":"a1b2c3d4e5f60005"
                    }
                 ]
              },
              "tape-link-fulfillment-state":"pending",
              "user-name":"SYSPROG"
        "tape-link-name":"first-tape",
        "tape-link-uri":"/api/tape-links/1d910e16-5f0e-11eb-856c-fa163edda9b5"
     },
     {
        "cpc-uri":"/api/cpcs/ea21f012-7597-3f35-a091-6d94577d5f50",
        "tape-link-actions":[
           {
              "action":"creation-requested",
              "action-time":1611580130447,
              "tape-link-configuration":{
                 "new-connectivity":2,
                 "new-description":"",
                 "new-max-partitions":1,
                 "new-name":"second-tape",
                 "tape-library-uri":null,
                 "world-wide-port-names-added":[
```


Figure 383. Get Tape Link Histories: Response (Part 3)

```
                        {
                           "status":"not-validated",
                           "world-wide-port-name":"a1b2c3d4e5f60006"
                        },
                        {
                           "status":"not-validated",
                           "world-wide-port-name":"a1b2c3d4e5f60005"
                        }
                     ]
                  },
                  "tape-link-fulfillment-state":"pending",
                  "user-name":"SYSPROG"
               },
               {
                  "action":"modification-requested",
                  "action-time":1611580150925,
                  "tape-link-configuration":{
                     "new-connectivity":3,
                     "old-connectivity":2,
                     "world-wide-port-names-added":[
                        {
                           "status":"not-validated",
                           "world-wide-port-name":"a1b2c3d4e5f60007"
                        }
                     ]
                  },
                  "tape-link-fulfillment-state":"pending",
                  "user-name":"SYSPROG"
               }
            ],
            "tape-link-name":"second-tape",
            "tape-link-uri":"/api/tape-links/76a8dce0-5f0e-11eb-bcd5-fa163edda9b5"
         }
      ]
   }
```


Figure 384. Get Tape Link Histories: Response (Part 4)

Update Tape Link Environment Report

The `Update Tape Link Environment Report` operation triggers asynchronous storage discovery for a tape link. The system will check the configuration for the tape link and create an environment report indicating the status of the configuration. Because the system periodically performs storage discovery for tape link and creates an environment configuration report for those tape links, it is not necessary to use this operation; however, issuing this operation may cause the discovery to be performed sooner than it would have otherwise.

HTTP method and URI

```
POST /api/tape-links/{tape-link-id}/operations/update-tape-link-environment-report
```

In this request, the URI variable *{tape-link-id}* is the object ID of the target tape link.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Table 330.			
Field name	**Type**	**Rqd/Opt**	**Description**
`force-restart`	Boolean	Optional	Indicates if there is an in-progress discovery operation for the specified tape link, it should be terminated and started again. If **false** and there is an in-progress discovery operation for the specified tape link, no new discovery is started. Default value: **false**

Response body contents

Once the request is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (**fields job-status-code** and **job-reason-code**) which are set as indicated in Table 332 on page 728. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation triggers detection of the tape drives (logical units or LUNs) for the WWPNs that are assigned to this tape link. Once the LUN discovery operation is completed, a report is generated for the tape link, indicating the status of the environment configuration. The Get Tape Link Environment Report operation can be used to retrieve it. The API user must have action/task permission to the **Configure Storage - System Programmer** task or the **Configure Storage - Storage Administrator** task; otherwise, status code 403 (Forbidden) is returned. A 404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.
- Action/task permission to the **Configure Storage – System Programmer** or the **Configure Storage - Storage Administrator** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in “Response body contents” on page 727.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 331. Update Tape Link Environment Report: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage - Storage Administrator** tasks.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	1	The state of the CPC to which the tape link is associated is not valid to perform the operations. It must be one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Table 332. Update Tape Link Environment Report: Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK)	N/A	The operation completed successfully.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the HMC or the API user does not have object-access permission for it.
500 (Server Error)	263	The update tape link environment report operation failed and there will be no automatic retry.

Example HTTP interaction

```
POST /api/tape-links/ed5f229c-5700-11eb-b4b6-00106f0d81c9/operations/
  update-tape-link-environment-report HTTP/1.1
x-api-session: 2ing9v6hkdfb6sytlg7jq510iawu064iafiaymo5jlvlmbjhv1 content-type: application/json
```

Figure 385. Update Tape Link Environment Report: Request

```
202 Accepted
server: Hardware management console API web server / 2.0
location: /api/jobs/0f5cb98a-5716-11eb-8353-00106f23d08c
cache-control: no-cache
date: Fri, 15 Jan 2021 09:43:02 GMT
content-type: application/json;charset=UTF-8
content-length: 60
{
   "job-uri":"/api/jobs/0f5cb98a-5716-11eb-8353-00106f23d08c"
}
```

Figure 386. Update Tape Link Environment Report: Response (Part 1)

Usage notes

- At any point in time, there will be at most one in-progress discovery operation checking configuration for a given tape link. While a discovery operation is in progress, any new request for the same tape link from the APIs or the Graphical User Interface will be linked to that in-progress discovery operation. An in-progress discovery operation can be interrupted for various reasons, such as starting a partition to which this tape link is attached, modifying adapters of the virtual resources of the tape link, or invoking the `Update Tape Link Environment Report` operation with a **force-restart** flag set to **true**. If an in-progress discovery operation is interrupted for any reason, the operation will be automatically retried, and the previous requests associated with that in-progress discovery operation will be linked to the new operation.
- If the request fails for any reason, all requests associated with that in-progress request will fail and an error code will be returned in the asynchronous result.
- If the HMC stops communicating with the Support Element or if the CPC enters into an invalid state, when the discovery operation for a tape link is in progress, the operation will be retried when the HMC starts communicating with the SE again or if the CPC enters a valid state, respectively. The valid CPC states for the discovery operation are **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.

Get Tape Link Environment Report

The `Get Tape Link Environment Report` operation retrieves the most recent environment configuration for a tape ink. The environment report for a tape link is first generated when a tape discovery operation completes successfully for the first time

HTTP method and URI

```
GET /api/tape-links/{tape-link-id}/operations/get-tape-link-environment-report
```

In this request, the URI variable *{tape-link-id}* is the object ID of the Tape Link object.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`last-scan-time`	Timestamp	The time the last discovery operation for a tape link completed successfully.
`cpc-name`	String	The name of the system.

Field name	Type	Description
`fabrics`	Array of fabric-info objects	The list of information about the fabrics in the tape link. Each element in the array is an instance of a fabric-info object, described in Table 333 on page 730. The list is empty if the configuration is still pending from the storage admin.
`number-of-tape-drives`	Integer	Total number of distinct tape drives sensed by all WWPNs across all adapters of the tape link.

Each fabric-info object contains the following fields:

Table 333. fabric-info nested object

Name	Type	Description
`fabric-id`	String (16)	A 16-character lower-case hexadecimal string that contains the World Wide Name (WWN) of the uplink Fibre Channel Switch.
`world-wide-port-names`	Array of world-wide-port-name-zone-info object	The list of information about all WWPNs zoned on this fabric. Each element in the array is an instance of a world-wide-port-name-zone-info object described in the next table.

Each nested world-wide-port-name-zone-info object contains the following fields:

Table 334. world-wide-port-name-zone-info nested object

Name	Type	Description
`world-wide-port-name`	String (16)	A 16-character lower-case hexadecimal string that contains a WWPN that uniquely identifies a host or initiator device in the SAN.
`zoned-adapters`	Array of adapter-info objects	The list of information about all adapters zoned for this WWPN in the SAN. Each element in the array is an instance of an adapter-info object described in the next table.
`zoning-status`	String Enum	The zoning status of the WWPN in the tape link. Values: • **"pending"** - Indicates that the zoning is pending. • **"complete"** - Indicates that the zoning is complete. • **"incomplete"** - Indicates that there is an error with zoning.

Each nested adapter-info object contains the following fields:

Table 335. adapter-info nested object

Name	Type	Description
`adapter-uri`	String/ URI	The canonical URI path of an adapter of the tape link.
`drawer`	Integer	The drawer number where the adapter is located

Table 335. adapter-info nested object (continued)		
Name	**Type**	**Description**
`tape-libraries`	Array of type-library-info objects	The list of tape libraries accessible through this adapter. Each element in the array is an instance of a tape-library-info object described in the next table. This value will be **null**, if a tape library has not been discovered yet and if a tape library was not specified when creating the tape link.
`config-status`	Array of String Enum	The configuration status of the adapter in the tape link. Values: • **"pending-with-mismatches"**- This adapter is connected to the selected tape library, but was not listed in the tape link request. If the **config-status** value for any adapter is **"pending-with-mismatches"**, the value of the tape link's **fulfillment-state** property will also be **"pending-with-mismatches"**. • **"degraded"** – Indicates that the adapter is degraded. • **"link-error"** – Indicates that there is a cable or optics error detected for the physical connection to the fabric • **"complete"** - Indicates that the adapter has been zoned.

Each nested tape-library-info object contains the following fields:

Table 336. tape-library-info nested object		
Name	**Type**	**Description**
`tape-library-uri`	String/ URI	The Canonical URI path (**object-uri**) of the Tape Library object. This value will be **null** when there is no control path found to the tape drives.
`tape-drive-identifiers`	Array of String	List of identifiers assigned by the vendor that uniquely identifies the tape drives of the corresponding tape library.
`config-status`	String Enum	The configuration status of the library. Values: • **"multiple-libraries-discovered"** - More than one library was discovered for the requested tape link. • **"unknown-library-discovered"** - Drives were sensed that could not be associated with a library. The **tape-library-uri** will be **null**. • **"different-library-discovered"** - A tape library was discovered that was not requested for the tape link. • **"complete"** - The requested tape library has been discovered. No library was requested and exactly one library was discovered. • **"library-not-reachable"** - The tape library can no longer be reached due to an error with cables or with the library itself.

Description

This operation retrieves the snapshot of the configuration of the specified tape link that was captured during the last discovery cycle.

The API user must have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage – Storage Administrator** tasks; otherwise, status code 403 (Forbidden) is returned. A

404 (Not Found) status code is returned if the object ID *{tape-link-id}* does not identify a Tape Link object to which the API user has object-access permission.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the tape link whose **object-id** is *{tape-link-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 729.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 337. Get Tape Link Environment Report: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Configure Storage – System Programmer** or **Configure Storage - Storage Administrator** tasks.
404 (Not Found)	1	A tape link with the **object-id** *{tape-link-id}* does not exist on the HMC or the API user does not have object-access permission for it.
409 (Conflict)	250	An environment report has not been generated.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/tape-links/ccb71f28-3aee-11eb-ad86-00106f0d81c9/operations/
  get-tape-link-environment-report HTTP/1.1
x-api-session: 3m8mqndvfwl7vtosroqs1b0p6rzxszc6voflw99rllr913oiw3
```

Figure 387. Get Tape Link Environment Report: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 15 Jan 2021 09:46:58 GMT
content-type: application/json;charset=UTF-8
content-length: 2262
{
   "fabrics":[
      {
         "fabric-id":"100000051E4A8F00",
         "world-wide-port-names":[
            {
               "world-wide-port-name":"c05076ffe8000017",
               "zoned-adapters":[
                  {
                     "config-status":[
                        "complete"
                     ],
                     "adapter-uri":"/api/adapters/ab61e64e-4fc4-11e9-b8fd-00106f0d81c9",
                     "drawer":2,
                     "tape-libraries":[
                        {
                           "config-status":"complete",
                           "tape-drive-identifiers":[
                              "IBM ULT3580-HH6 1068012586",
                              "IBM ULT3580-HH6 90WT800375"
                           ],
                           "tape-library-uri":"/api/tape-libraries/
                             b6303362-39e4-11eb-b4fe-00106f0d81c9"
}
                     ]
                  },
                  {
                     "config-status":[
                        "complete"
                     ],
                     "adapter-uri":"/api/adapters/ab61e64e-4fc4-11e9-b8fd-00106f0d81c9",
                     "drawer":2,
                     "tape-libraries":[
                        {
                           "config-status":"complete",
                           "tape-drive-identifiers":[
                              "IBM ULT3580-HH6 1068012586",
                              "IBM ULT3580-HH6 90WT800375"
                           ],
                           "tape-library-uri":"/api/tape-libraries/
                             b6303362-39e4-11eb-b4fe-00106f0d81c9"
                        }
                     ]
                  }
               ],
               "zoning-status":"complete"
            },
```

Figure 388. Get Tape Link Environment Report: Response (Part 1)

```
            {
               "world-wide-port-name":"c05076ffe8000017",
               "zoned-adapters":[
                  {
                     "config-status":[
                        "complete"
                     ],
                     "adapter-uri":"/api/adapters/ab61e64e-4fc4-11e9-b8fd-00106f0d81c9",
                     "drawer":2,
                     "tape-libraries":[
                        {
                           "config-status":"complete",
                           "tape-drive-identifiers":[
                              "IBM ULT3580-HH6 1068012586",
                              "IBM ULT3580-HH6 90WT800375"
                           ],
                           "tape-library-uri":"/api/tape-libraries/
                             b6303362-39e4-11eb-b4fe-00106f0d81c9"
                        }
                     ]
                  },
                  {
                     "config-status":[
                        "complete"
],
                     "adapter-uri":"/api/adapters/ab61e64e-4fc4-11e9-b8fd-00106f0d81c9",
                     "drawer":2,
                     "tape-libraries":[
                        {
                           "config-status":"complete",
                           "tape-drive-identifiers":[
                              "IBM ULT3580-HH6 1068012586",
                              "IBM ULT3580-HH6 90WT800375"
                           ],
                           "tape-library-uri":"/api/tape-libraries/
                             b6303362-39e4-11eb-b4fe-00106f0d81c9"
                        }
                     ]
                  }
               ],
               "zoning-status":"complete"
            }
         ]
      }
   ],
   "last-scan-time":1610636246000,
   "number-of-tape-drives":2,
   "cpc-name":"P0000M12"
}
```


Figure 389. Get Tape Link Environment Report: Response (Part 2)

Usage note

If an environment report has not been generated for a tape link, use the `Update Tape Link Environment Report` operation to initiate the creation of the environment report.

Inventory service data

Information about the Tape Links managed by the HMC can be optionally included in the inventory data provided by the Inventory Service

Inventory entries for Tape Link objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of **class "tape-link"** are to be included. Information for a particular tape link is included only if the API user has object-access permission to that object.

For each tape link to be included, the inventory response array includes the following:

- An array entry for the tape link object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for “Get Tape Link Properties” on page 689. That is,

the data provided is the same as would be provided if a `Get Tape Link Properties` operation were requested targeting this object.

- An array entry for each virtual tape resource element associated with the tape link. For each such virtual tape resource, an entry is included that is a JSON object with the same contents as is specified in the Response body contents section for "Get Virtual Tape Resource Properties" on page 709.

Sample inventory data

The following fragment is an example of the JSON objects that would be included in the `Get Inventory` response to describe a storage template. These objects would appear as multiple array entries in the response array:


```
{
   "adapter-port-uris":[
      "/api/adapters/4f8da200-6574-11eb-a6e2-fa163e65c83b/storage-ports/0",
      "/api/adapters/92ff8ef4-6574-11eb-8158-fa163e65c83b/storage-ports/0"
   ],
   "class":"tape-link",
   "connectivity":2,
   "cpc-uri":"/api/cpcs/d629c0a5-80cf-3590-b7f0-2456edc6652d",
   "description":"",
   "fulfillment-state":"complete",
   "max-partitions":1,
   "name":"A tape link",
   "object-id":"4c57bc36-6661-11eb-abd9-fa163e65c83b",
   "object-uri":"/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b",
   "parent":"/api/console",
   "tape-library-uri":"/api/tape-libraries/4f3400a4-6661-11eb-86b8-fa163e65c83b",
   "unassigned-world-wide-port-names":[],
   "virtual-tape-resource-uris":[
      "/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b/virtual-tape-resources/
c92078e0-6669-11eb-a93e-fa163e65c83b",
      "/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b/virtual-tape-resources/
c8d7a91c-6669-11eb-a93e-fa163e65c83b"
   ]
},
{
   "adapter-port-uri":"/api/adapters/92ff8ef4-6574-11eb-8158-fa163e65c83b/storage-ports/0",
   "class":"virtual-tape-resource",
   "degraded-reasons":[],
   "description":"",
   "device-number":"0002",
   "element-id":"c92078e0-6669-11eb-a93e-fa163e65c83b",
   "element-uri":"/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b/virtual-tape-resources/
c92078e0-6669-11eb-a93e-fa163e65c83b",
   "name":"vhba_TL_A tape link1",
   "parent":"/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b",
   "partition-uri":"/api/partitions/c7cfd99a-6669-11eb-a93e-fa163e65c83b",
   "world-wide-port-name-info":{
      "status":"validated",
      "world-wide-port-name":"a1b2c3d4e5f60003"
   }
},
{
   "adapter-port-uri":"/api/adapters/4f8da200-6574-11eb-a6e2-fa163e65c83b/storage-ports/0",
   "class":"virtual-tape-resource",
   "degraded-reasons":[],
   "description":"",
   "device-number":"0001",
   "element-id":"c8d7a91c-6669-11eb-a93e-fa163e65c83b",
   "element-uri":"/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b/virtual-tape-resources/
c8d7a91c-6669-11eb-a93e-fa163e65c83b",
   "name":"vhba_TL_A tape link0",
   "parent":"/api/tape-links/4c57bc36-6661-11eb-abd9-fa163e65c83b",
   "partition-uri":"/api/partitions/c7cfd99a-6669-11eb-a93e-fa163e65c83b",
   "world-wide-port-name-info":{
      "status":"validated",
      "world-wide-port-name":"a1b2c3d4e5f60002"
   }
}
```


Figure 390. Tape Link object: Sample inventory data - Response

Partition Link object

Support Element version 2.16.0 introduced the concept of Partition Links. A Partition Link interconnects two or more partitions that share the same network configuration and (currently) reside on the same DPM-enabled CPC. Using Partition Links, you can quickly configure network connections among partitions on the same system to improve performance.

Types of Partition Links:

Partition Links are based on different technologies, support for a specific technology is determined by the availability of the corresponding API feature:

- **dpm-smcd-partition-link-management**: indicates that the CPC supports Partition Links of **type "smc-d"**, which enable communication between partitions on the same system through the Shared Memory Communications - Direct Memory Access (SMC-D) (Version 2 or later) technology. This technology provides high-bandwidth, low-latency TCP/IP traffic over internal shared memory (ISM) devices for improved performance. Albeit the corresponding feature is exposed as firmware and as API feature, it is recommended to rely on the availability of the API feature. [Added by feature **dpm-smcd-partition-link-management**]
- **dpm-hipersockets-partition-link-management**: indicates that the CPC supports Partition Links of **type "hipersockets"**, which enable high-speed TCP/IP connectivity between partitions on the same system through the system memory of the CPC. Therefore there is no need for any physical cabling/adapters. The HiperSockets implementation is based on the OSA-Express Queued Direct I/O (QDIO) protocol. [Added by feature **dpm-hipersockets-partition-link-management**]
- **dpm-ctc-partition-link-management**: indicates that the CPC supports Partition Links of **type "ctc"**, which are based on FICON Channel to Channel interconnect technology. This requires the corresponding cabling between FICON adapters (switched or point to point) to be in place. Currently the implementation is limited on interconnecting partitions residing on the same CPC. [Added by feature **dpm-ctc-partition-link-management**]

Partition Link states:

A Partition Link has a **state** property that indicates its current state related to the partitions sharing the link. The following states are supported:

- **"complete"** - All partitions of this partition link are connected.
- **"incomplete"** - Less than 2 partitions are connected to this partition link.
- **"updating"** - Partitions are currently being added or removed from the partition link.

Data model

This object includes the properties that are defined in the "Base managed object properties schema" on page 100, with the following class-specific specializations:

Table 338. Partition Link object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Partition Link object is of the form `/api/partition-links/{partition-link-id}` where `{partition-link-id}` is the value of the **object-id** property of the Partition Link object.
parent	—	String/ URI	The parent of a partition link is conceptually its owning Console, and so the parent value is the canonical URI path for the Console.
class	—	String (14)	The class of a Partition Link object is **"partition-link"**.

Table 338. Partition Link object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(w)(pc)	String (1-64)	The display name specified for the partition link. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique to the other partition links associated to the same CPC.
description	(w)(pc)	String (0-200)	The description associated with this partition link. Default value: An empty string

Class specific additional properties

In addition to the properties defined in the base managed object, this object includes the following additional class-specific properties:

Note: Certain properties are only valid for partition links of a specific **"type"**. These values are only included in a Partition Link object if the partition link is of that type. For example a partition link with **type "smc-d"** will define a **starting-fid** property (**"smc-d"**only) and **bus-connections** property (**"smc-d"**, **"hipersockets"**) but not a **"starting-device-number"** property (**"hipersockets"** only).

Table 339. Partition Link object: class specific properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
type	—	String Enum	Defines the type of the partition link. One of the following values: • **"smc-d"** - A SMC-D (ISM) link. • **"hipersockets"** - A Hipersockets (IQD) link. [Added by feature **dpm-hipersockets-partition-link-management**] • **"ctc"** - A FICON CTC link. [Added by feature **dpm-ctc-partition-link-management**]	All
state	(pc)	String Enum	The current state of the partition link. Values: • **"complete"** - All partitions of this partition link are connected. • **"incomplete"** - Less than 2 partitions are connected to this partition link or an adapter is in degraded state. • **"updating"** - Partitions are currently being added or removed from this partition link.	All
cpc-uri	—	String/ URI	The canonical URI path of the CPC object associated with this partition link.	All
cpc-name	(a)	String (1-64)	The name of the CPC object to which the Partition Link is associated.	All
pending-operations	(pc)(c)	Array of pending-operations objects	Describes the pending operations information associated with the partitions link. Default: an empty array	All

Table 339. Partition Link object: class specific properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
incomplete-reasons	—	Array of String Enum	An array of reasons for a partition link being in an **"incomplete"** state. Incomplete reasons for a partition link: • **"adapter-degraded"** - One of the adapters used in the paths of a CTC partition link is degraded. • **"insufficient-partitions"** - The minimum number of partitions for a partition link is not met.	All
starting-fid	(w)(pc)	Integer	Starting number for the Functional ID assignment. If no FID for a NIC is provided, the next available FID starting from **starting-fid** will be used. Constraint: The value must be within the range defined by the values of the **minimum-fid-number** and **maximum-fid-number** properties of the hosting CPC. Default: 4096	smc-d
adapter-uri	—	String/ URI	The canonical URI path of the backing Adapter object.	smc-d, hipersockets
bus-connections	(pc)(c)	Array of bus-connection objects	Devices associated with the partition link object. Constraint: Each partition may only have a single element in the array. Default: an empty array	smc-d, hipersockets
maximum-transmission-unit-size	(w)(pc)	Integer	The maximum transmission unit size of the HiperSockets adapter. The maximum frame size is implied by this value. Values: • **8** - 8 KB MTU size and 16 KB maximum frame size • **16** - 16 KB MTU size and 24 KB maximum frame size • **32** - 32 KB MTU size and 40 KB maximum frame size • **56** - 56 KB MTU size and 64 KB maximum frame size. Default: 8	hipersockets
starting-device-number	(w)(pc)	String (4)	Starting number for the device number assignment. If no device number for a NIC is provided, the next available device number starting from starting-device-number will be used. The valid range is 0001-FFFF. Default: 7400	hipersockets
devices-per-path	(w)(pc)	Integer (1-16)	The number of devices between any partition pair (or endpoint-pair) in every path of a partition link. Default: 4	ctc
paths	(w)(pc)(c)	Array of ctc-path-details objects	An array of physical paths used for communication between selected partitions. An array of one or more paths represented by the ctc-path-details nested object, where each element defines property values of a physical path used for communication between selected partitions.	ctc

Table 340. pending-operations nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
partition-uri	—	String/ URI	The URI of the partition that has pending operations. **Null** if the API user does not have object-access permission to the partition.	All
operation	—	String Enum	The type of the pending operation object. Values: • **"attach"** - The partition is pending to be attached to the partition link. • **"detach"** - The partition link is pending to be detached from the partition link.	All
partition-name	(a)	String	The name of the partition that has pending operations.	All

ctc: Nested object properties

Table 341. ctc-path-details nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
devices	(w)	Array of ctc-endpoints-details objects	Each element specifies the devices owned by all partitions mentioned in each endpoint-pair.	ctc
starting-device-number	(w)	String (4)	The starting offset for the device number generations on all requested partitions in this path. This is a string in the form of a 4-digit hexadecimal number. The allowed value range is from 0001-FFFF.	ctc
adapter-port-info	(w)	adapter-info object	Describes the information about the FICON Adapter object that represents the adapter port of the path.	ctc
connecting-adapter-port-info	(w)	adapter-info object	Describes the information about the Adapter object that represents the connecting adapter port of the path.	ctc

Table 342. ctc-endpoints-details nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
endpoint-pair	(w)	Array of ctc-partition-device-endpoint-details objects	Each element of array specifies the device numbers used in creating CTC devices in the partitions associated with the partition link.	ctc

Table 343. adapter-info nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
adapter-uri	(a)	String/ URI	Canonical URI path of the FICON Adapter object. Null if user does not have access to the Adapter.	ctc
adapter-name	(a)	String	The **name** property of the FICON Adapter object.	ctc
adapter-status	(a)	String Enum	The **status** property of the FICON Adapter object.	ctc
fabric-name	(a)	String	The name of the fabric associated with the storage switch to which the FICON Adapter is connected. It will be **null** for a **point-to-point** path.	ctc
switch-name	(a)	String	The display name specified for the storage switch. The length and character requirements on this property are the same as those of the name property described in the "Base managed object properties schema" on page 100 Names must be unique to the other storage sites within the FICON configuration of its containing storage site and storage fabric. Default value: Currently of the for "Storage switch *{switch-domain-id}*", where *{switch-domain-id}* is the value of the **switch-domain-id** property	ctc
switch-domain-id	(a)	String	A two-character hexadecimal number that represents the domain identifier assigned to the storage switch to which the FICON Adapter is connected. it will be **null** for a **point-to-point** path.	ctc

Table 344. ctc-partition-device-endpoint-details nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
partition-uri	(w)	String/ URI	Canonical URI path of the Partition object. Null if user does not have access to the partition.	ctc
device-numbers	(w)	Array of String (4)	Each element in the array specifies the device number assigned to the CTC device owned by the partition represented by **partition-uri**. Empty array if the API user does not have object-access permission to the partition.	ctc
partition-name	(a)	String	The name of the partition.	ctc

smc-d/hipersockets: Nested object properties

Table 345. bus-connection nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
partition-uri	(w)	String/ URI	The URI of the partition to which this bus connection attaches. **Null** if the API user does not have object-access permission to the partition.	smc-d, hipersockets
nics	(w)	Array of NIC objects	A list of NIC objects owned by the specified partition. Empty array if the API user does not have object-access permission to the partition.	smc-d, hipersockets

Table 345. bus-connection nested object properties (continued)				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
partition-name	(a)	String	The name of the partition to which this bus connection attaches.	smc-d, hipersockets

Table 346. nic nested object properties				
Name	**Qualifier**	**Type**	**Description**	**Supported "type" values**
device-numbers	(w)	Array of String	The value to be set as the NIC's **device-numbers** property. Each device number is a string in the form of a 4-digit hexadecimal number. If the **type** of the partition link is **"hipersockets"** a set of up to 3 consecutive device numbers can be specified and the allowed value range is from 0000-FFFF. If the **type** of the partition link is **"smc-d"**, only one device number is allowed and the value range is from 0001-FFFF. Default: auto-generated. Constraint: If the **type** of the partition link is **"smc-d"**, this number must be unique across the device numbers of all other NIC elements of **type "roce"** or **"cna"** and all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition. If the **type** is **"hipersockets"**, the number(s) must be unique across the device numbers of all other partition links of **type "hipersockets"** and all instances of objects listed in "Channel-based device numbers" on page 197 of the partition.	smc-d, hipersockets
fid	(w)	Integer	Functional ID of the NIC. Only valid when **type** is **smc-d**. The FID must be unique across all FIDs defined on the hosting CPC. Default: auto-generated Constraint: The value must be within the range defined by the values of the **minimum-fid-number** and **maximum-fid-number** properties of the hosting CPC.	smc-d
uuid	—	String (36)	UUID representing the NIC element object owned by a partition.	smc-d, hipersockets
vlan-id	(w)	Integer (1-4094)	The VLAN ID associated with this NIC. This value can only be set when the **type** of the link is **"hipersockets"**.	hipersockets
mac-address	(w)	String	The MAC address associated with this NIC. It must be unique among all the NICs created in the CPC. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addressed are valid. For example, "02:ff:12:34:56:78". This value can only be set when the **type** of the link is **"hipersockets"**. Default: auto-generated.	hipersockets
is-boot-network-device	(a)	Boolean	This flag indicates whether this NIC is used a boot network device of the partition. This flag will only be returned when the **type** of the link is **"hipersockets"**.	hipersockets

Create Partition Link

The `Create Partition Link` operation creates a new partition link object with the given properties.

HTTP method and URI

```
POST /api/partition-links
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description	Supported "type" values
`name`	String (1-64)	Required	The value to be set as the partition link's **name** property.	All
`description`	String (0-200)	Optional	The value to be set as the partition link's **description** property. Default: an empty string	All
`type`	String Enum	Required	The value to be set as the partition link's **type** property.	All
`cpc-uri`	String/ URI	Required	The value to be set as the partition link's **cpc-uri** property.	All
`bus-connections`	Array of new-bus-connection objects	Optional	The value to be set as the partition link's **bus-connections** property.	smc-d, hipersockets
`starting-fid`	Integer	Optional	The value to be set as the partition link's **starting-fid** property. Default: 4096	smc-d
`starting-device-number`	String (4)	Optional	The value to be set as the partition link's **starting-device-number** property. Default: 7400	hipersockets
`maximum-transmission-unit-size`	Integer	Optional	The value to be set as the partition link's **maximum-transmission-unit-size** property. Only valid for the link **type "hipersockets"**. Default: 8	hipersockets
`partitions`	Array of String/ URI	Required if **type** is **"ctc"**	A list of canonical URIs representing Partition objects to be added to the partition link.	ctc
`paths`	Array of added-ctc-path-info objects	Required if **type** is **"ctc"**	An array of one or more paths represented by added-ctc-path-info nested objects where each element defines a new physical path that is to be used for communicating between provided partitions.	ctc

Field name	Type	Rqd/Opt	Description	Supported "type" values
`devices-per-path`	Integer (1-16)	Optional	Specifies the number of devices to be created per path on each partition provided in the request. Constraint: **devices** is an optional field. If provided, the number of device numbers provided in **device-numbers** field must always be equal to the **devices-per-path** count. Default: 4	ctc

smc-d/hipersockets: Nested object properties

Each nested new-bus-connection object contains the following fields:

Table 347. new-bus-connection nested object properties

Name	Type	Rqd/Opt	Description	Supported "type" values
`partition-uri`	String/ URI	Required	The canonical URI of the Partition object to be attached to the partition link.	smc-d, hipersockets
`number-of-nics`	Integer	Required	Specifies how many NICs should be created. Additional details such as device numbers or FIDs of the newly created NICs can be specified with the **nics** field. • When **type** is **"smc-d"**, valid values are 1-255. • When **type** is **"hipersockets"**, valid values are 1-4096.	smc-d, hipersockets
`nics`	Array of new-nic objects	Optional	An array of NICs to be added to the specified partition connecting to the newly created partition link. Constraint: The length of the array must be less than or equal to the value of the **number-of-nics** field.	smc-d, hipersockets

Each nested new-nic object contains the following fields:

Table 348. new-nic nested object properties				
Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`device-numbers`	Array of String (4)	Optional	The value to be set as the NIC's **device-numbers** property. Each device number is a string in the form of a 4-digit hexadecimal number. If the **type** of the partition link is **"hipersockets"**, a set of up to 3 consecutive device numbers can be specified and the allowed value range is from 0000-FFFF. If the **type** of the partition link is **"smc-d"**, only one device number is allowed and the value range is from 0001-FFFF. Default: auto-generated. Constraint: If the **type** of the partition link is **"smc-d"**, this number must be unique across the device numbers of all other NIC elements of type **roce** or **cna** and all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition. If the **type** is **"hipersockets"**, the number(s) must be unique across the device numbers of all other partition links of **type "hipersockets"** and all instances of object listed in "Channel-based device numbers" on page 197 of the partition	smc-d, hipersockets
`fid`	Integer	Optional	The value to be set as the NIC's **fid** property. Constraints: • The value must be unique across all FIDs defined on the same CPC. • The value must be within the range defined by the values of the **minimum-fid-number** and **maximum-fid-number** properties of the hosting CPC. Default: auto-generated	smc-d
`vlan-id`	Integer (1-4094)	Optional	The VLAN ID associated with this NIC. This value can only be set when the **type** of the link is **"hipersockets"**.	hipersockets
`mac-address`	String	Optional	The MNAC address associated with this NIC. It must be unique among all the NICs created in the CPC. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addresses are value, for example, "02:ff:12:34:56:78". This value can only be set when the **type** of the link is **"hipersockets"**. Default: auto-generated	hipersockets

ctc: Nested object properties

Each nested added-ctc-path-info object contains the following fields:

Table 349. added-ctc-path-info nested object properties				
Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`starting-device-number`	String (4)	Optional	The value to be set as the path's **starting-device-number** property. The **starting-device-number** is a string in the form of a 4-digit hexadecimal number. The allowed value range is from 0000-FFFF. If not provided, the value will be automatically generated. The default value is 4000.	ctc
`devices`	Array of ctc-endpoint objects	Optional	Each element specifies the devices owned by all associated partitions in the partition link. An array of **ctc-endpoint** objects, each **ctc-endpoint** has a pair of endpoints between two partitions to form a valid communication path. Adding a new path: • It is optional to provide **devices** for a new path. The **devices** will be automatically generated for all partitions associated with the partition link if no **devices** are provided in the request. • It is allowed to provide **devices** for some of the partition combinations and **devices** are automatically generated for the rest of the partitions associated with the partition link. The number of devices to be provided in **device-numbers** must be equal to **devices-per-path** provided in the request. The **endpoint-pair** array must always have 2 entries representing a connection between a partition pair. Constraint: **devices** is an optional field. If provided, the number of device numbers provided in the **device-numbers** field must always be equal to the **devices-per-path** count.	ctc
`adapter-port-uri`	String/URI	Required	Canonical URI path of the FICON Adapter object that represents the adapter port of the path.	ctc
`connecting-adapter-port-uri`	String/URI	Required	Canonical URI path of the FICON Adapter object that represents the connecting adapter port of the path.	ctc

Each nested ctc-endpoint object contains the following fields:

Table 350. ctc-endpoint nested object properties				
Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`endpoint-pair`	Array of ctc-partition-devices-endpoint objects	Required	Each element of the array specifies the device numbers to be used in creating ctc devices for a partition associated with the Partition link. Constraints: • Each endpoint pair must have exactly two instances of ctc-partition-devices-endpoint objects to form a valid communication path between provided partitions in the endpoint pair. And each endpoint must have a different **partition-uri** in the endpoint pair combination. • It is optional to provide **endpoint-pair** for any partition combination. If **endpoint-pair** is provided in the request, the **partition-uri** and **device-numbers** properties are required for both endpoints in an endpoint pair.	ctc

Each nested ctc-partition-devices-endpoint object contains the following fields:

Table 351. ctc-partition-devices-endpoint nested object properties				
Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`device-numbers`	Array of String (4)	Required	Each device number in the **device-numbers** property is a string in the form of a 4-digit hexadecimal number. The allowed value range is from 0000=FFFF. Constraints: • If the **type** of the partition link is **"ctc"**, the number(s) must be unique across the device numbers of all other partition links of **type "ctc"**, and all instances of objects listed in "Channel-based device numbers" on page 197 of the partition • The number of device numbers provided in this property must always be equal to the **devices-per-path** count.	ctc
`partition-uri`	String/ URI	Required	The canonical URI representing a partition to be added to the partition link.	ctc

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates of the asynchronous operation on the Partition link.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for "Query Job Status" on page 151. When the status of the job is **"complete"**, the results include a job

completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 754. When this operation is successful, the **job-results** field contains an object with the following fields:

Field Name	Type	Description
`operation-results`	Array of attach-operation-status objects	Array of object in which each object represents the attachment status of a partition specified in the request body.

Each nested attach-operation-status object contains the following fields:

Field Name	Type	Description
`partition-uri`	String/ URI	The canonical URI path of the Partition object that is being added or removed.
`operation-status`	String Enum	The status of the asynchronous attach operation: • **"attached"** - The partition is successfully attached. • **"pending-retry"** - The attach operation failed in the first attempt. The operation will be retried with the next SE restart. • **"failed"** - The attach operation failed and the requested operation is canceled. One possible scenario could be that the requested partition no longer exists on the system.

When it is not successful, the **job-results** field contains an object with the following fields:

Table 352. Table 9. error-job-results nested object

Field Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation creates a partition link on the CPC identified by the **cpc-uri** field with the values specified in the request body. Any fields identified as required must be included in the request body. Any fields identified as optional may be excluded from the request body; if an optional field is not found in the request body, its value will be set to its default value. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

When the request body is valid, a 202 (Accepted) status code is sent as soon as the new partition link object has been created, but before the partitions are attached to it. The attachment of the individual partitions and the activation of the associated devices in each partition is performed asynchronously. The response has a Location header that provides the URI of the created partition link and the response body contains a URI that may be queried to retrieve the status of the asynchronous part of the operation. See "Query Job Status" on page 151 for information on how to query the job status. The operation always triggers an asynchronous job even if the request does not contain any partitions to be attached.

The newly created partition link stays in **"updating"** state until all requested partitions have been successfully attached to the partition link. If the attach operation for any partition fails due to an error, the partition link will stay in **"updating"** state and the failed operations will be retried after an SE reboot. If the retry for a partition is not successful, this partition will be removed from the partition link and a hardware message on the owning CPC is created. The asynchronous job completes after all partition attachments have either succeeded or failed once. The **job-results** property contains further details on the outcome of the individual attachments.

If the API user does not have action/task permission to the **Create Partition Link** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **cpc-uri** or any of the **partition-uri** fields do not designate a CPC or respectively a Partition object for which the API user has object-access permission. If the designated CPC is not in DPM mode, does not support the feature needed for the partition link type specified in the type field of the request, or is not in a valid state, 409(Conflict) status code is returned. A 409 (Conflict) status code is also returned if limits of the CPC or the partition are exceeded or values in the request body conflict with existing ones.

SMC-D and HiperSockets Partition Links

The connections between the partitions of the newly created partition link are described by the elements in the bus-connections array of the request body. Each entry in the bus-connections array must refer to an existing partition and define a set of NICs to be created. The NICs can be specified implicitly by providing only the **number-of-nics** field or optionally with additional details as part of the nics array. If the FID or device numbers are not specified for a SMC-D NIC, they will be automatically generated. The value of the **starting-fid** field will be used as the starting point for searching available FID numbers. If the device numbers or MAC address are not specific for a Hipersockets NIC, they will be automatically generated. The value of the **starting-device-number** field will be used as the starting point for searching available device numbers.

CTC Partition Links

The connections between the partitions are represented with in **paths** defined in the request. Each entry in the **paths** array represents a new CTC path to be defined through which the connections are formed between the partitions provided in the request. A valid CTC path can be defined by providing **adapter-port-uri** and **connecting-adapter-port-uri** representing FICON adapters enabled in CTC mode.

The defined path will be known as **switched** if both the provided adapters are connected to same switch, **switched-loopback** if both the provided adapters are representing same FICON Adapter, and **point-to-point** if both the provided adapters are connected to each other.

Each path can optionally have devices array representing the devices to be created on each partition provided in the request. Each entry in the devices array represent an **endpoint-pair.** It is optional to provide **endpoint-pair** in the request. If provided the number of **endpoint-pair** entries must be less than or equal to the total number of combinations possible between the partitions provided in partitions field in the request. Example: If 4 partitions are provided in partitions field, the number of **endpoint-pairs** entries must be 6. Partitions = {A,B,C,D},endpoint pairs = {(A,B),(A,C),(A,D),(B,C),(B,D),(C,D)}. Each entry in the **endpoint-pair** array represents a combination of partitions in a pair along with the devices represented by **device-numbers** to be created on each partition represented by **partition-uri**. The number of devices to be provided in the **device-numbers** must be equal to **devices-per-path** provided in the request. The **endpoint-pair** array must always have 2 entries representing a connection between a partition pair. If **endpoint-pair** entries are not provided in the request, the device numbers will be automatically generated for the partitions. The value of the **starting-device-number** field will be used as the starting point for searching available device numbers. The **starting-device-number** is automatically generated if not provided in the request.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-uri** is **cpc-uri**.
- Object-access permission to all Partition objects designated by a **partition-uri** field.
- Object-access permission to all FICON adapter objects designated by **adapter-port-uri** and **connecting-adapter-port-uri** fields.
- Action/task permission to the **Create Partition Link** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in “Response body contents” on page 746.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 353. Create Partition Link: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	4	For partition links of **type "hipersockets"**: • The API feature **dpm-hipersockets-partition-link-management** is not available on the CPC identified by the **cpc-uri** field. For partition links of **type "ctc"**: • The API feature **dpm-ctc-partition-link-management** is not available on the CPC identified by the **cpc-uri** field.
	8	The value of a field does not provide a unique value for the corresponding data model property as required. The partition link **name** provided by the user is already in use by another partition link associated with the hosting CPC. For partition links of **type "ctc"**: • The **partition-uri** combination in the **endpoint-pair** in the request must be unique.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body. A prerequisite condition or dependency among request body fields is not met. For partition links of **type "ctc"**: • The requested number of devices in the **device-numbers** must always be equal to the **devices-per-path** of the partition link. • The newly added **partition-uri** provided in the endpoint-pair of the request must be provided in the **partition-uris** for the request.
	452	For partition links of **type "smc-d"**: • The provided **device-numbers** of any of the NICs are already in use by an instance of one of the objects listed in "PCI-based device numbers" on page 197 of the partition. For partition links of **type "hipersockets"**: • The provided **device-numbers** of any of the NICs are already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197 of the partition. For partition links of **type "ctc"**: • The provided **device-numbers** of any of the NICs are already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197 of the partition. The **error-details** field of the response body contains a conflicting-device-numbers object (as described in Table 354 on page 753) listing the device number values that are not unique.

Table 353. Create Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	455	For partition links of **type "smc-d"**: • The provided **fid** of any of the NICs is already in use on the hosting CPC. The **error-details** field of the response body contains an invalid-fid-details object (as described in Table 356 on page 754) listing the FID values that are not unique.
	456	For partition links of **type "ctc"**, the provided combination of **adapter-uris** in **paths** does not form a valid path.
	458	For partition links of **type "ctc"**, one of the adapters provided in the **paths** is not configured in FICON CTC mode.
	459	For partition links of **type "ctc"**, the **paths** property must have at least one path defined to create the partition link.
	460	For partition links of **type "hipersockets"**, the provided **macs** of any of the NICs is already in use on the hosting CPC. The **error-details field** of the response body contains an **invalid-mac-details** object listing the MAC values that are not unique.
	462	For partition links of **type "ctc"**, the FICON adapter URI provided to define a path is already used by another path defined in the request.
403 (Forbidden)	1	API user does not have the required action/task permissions.
404 (Not Found)	2	A URI in the request body does not designate an existing resource of the expected type or designates a resource for which the user does not have object-access permission. The **error-details** field of the response body contains an not-found-details object defined in Table 360 on page 754, which identifies the objects that could not be found.

Table 353. Create Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	5	The CPC identified by the **cpc-uri** field is not enabled for DPM.
	6	The operation cannot be performed because the state of the CPC hosting the partition link is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	13	For partition links of **type "smc-d"**: • The CPC identified by the **cpc-uri** field does not support the **dpm-smcd-partition-link-management** feature.
	329	The operation cannot be performed because the CPC identified by the **cpc-uri** field is an unmanaged CPC, which is not supported by this operation.
	427	The CPC identified by the **cpc-uri** already has the maximum number of ISM or HiperSockets adapters defined.
	550	For partition links of **type "smc-d"**, the CPC identified by the **cpc-uri** already has the maximum number of FIDs defined.
	551	Unable to generate new device numbers as there are not enough free device numbers available.
	552	For partition links of **type "smc-d"**: • The provided FID of any of the NICs has reached the maximum allowed value on the CPC. The **error-details** field of the response body contains an invalid-fid-details object listing all the FIDs that are exceeding the maximum allowed value on the CPC.
	553	For partition links of **type "smc-d"**: • The starting FID value is beyond the allowed FID value range in the system.
	554	The maximum number of allowed partitions for a partition link has been reached.

Table 353. Create Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	556	For partition links of **type "smc-d"**: • The maximum number of Function ID (FIDs) per Internal Shared Memory (ISM) adapter that may be created for the CPC has been exceeded.
	557	For partition links of **type "hipersockets"**, the operation failed because it requires the generation of one or more MAC addresses, the but range of available addresses has been exhausted. [Added by feature **dpm-hipersockets-partition-link-management**]
	558	For partition links of **type "ctc"**, the number of devices defined in the partition for a path has exceeded the limit of 256. [Added by feature **dpm-ctc-partition-link-management**]
	561	For partition links of **type "hipersockets"**, the maximum number of HiperSockets devices that may be created for the CPC has been exceeded. [Added by feature **dpm-hipersockets-partition-link-management**]
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

The conflicting-device-numbers object contains the following fields:

Table 354. conflicting-device-numbers nested object		
Field Name	**Type**	**Description**
`conflicting-device-numbers`	Array of conflicting-device-info objects	List of all the conflicting devices associated with the partition object

The conflicting-device-info object contains the following fields:

Table 355. conflicting-device-info nested object		
Field Name	**Type**	**Description**
`partition-uri`	String/ URI	The URI of the partition corresponding to the conflicting device numbers.
`device-numbers`	Array of String	Array of conflicting device numbers of the partition.

The invalid-fid-details object contains the following fields:

Table 356. invalid-fid-details nested object		
Field Name	**Type**	**Description**
`fids`	Array of Integer	An array of FIDs that are invalid.

The invalid-path-error object contains the following fields:

Table 357. invalid-path-error nested object		
Field Name	**Type**	**Description**
`paths`	Array of adapter-uris objects	The value of this field contains all the paths that are invalid.

The adapter-uris object contains the following fields:

Table 358. adapter-uris nested object		
Field Name	**Type**	**Description**
`adapter-port-uri`	String/ URI	The canonical URI of the source adapter of the path.
`connecting-adapter-port-uri`	String/ URI	The canonical URI of the target adapter of the path.

The invalid-mac-details object contains the following fields:

Table 359. invalid-mac-details nested object		
Field Name	**Type**	**Description**
`macs`	Array of String	The value of this field contains all the MACs that are invalid.

The not-found-details object contains the following fields:

Table 360. not-found-details nested object		
Field Name	**Type**	**Description**
`object-uris`	Array of String/ URI	The list of URIs of objects (partition, partition links) that are not found.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 361. Create Partition Link: Job status and reason codes		
HTTP error status code	**Reason code**	**Description**
200 (OK)	N/A	The Asynchronous attach operation is completed. The detailed status of the operation is available in the **job-results** field in the response
500 (Server Error)	100	The job failed due to an internal error. The message text describes the error reason.
	101	Partition link asynchronous attach job timed out.

Table 361. Create Partition Link: Job status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Example HTTP interaction

```
POST /api/partition-links HTTP/1.1
x-api-session: 61cij6i72fadwx6xnr0tkf5tyr42yt9wke9173vgkjmwsnyr85
Content-Type: application/json
Content-Length: 374
{
   "bus-connections":[
      {
         "number-of-nics":2,
         "partition-uri":"/api/partitions/51b79be4-ac13-11ec-86a7-fa163eaa0f16"
      },
      {
         "number-of-nics":2,
         "partition-uri":"/api/partitions/9a417984-ac13-11ec-b798-fa163eaa0f16"
      }
   ],
   "cpc-uri":"/api/cpcs/a4aa2e69-9205-38ae-b55e-4d6ada6dea6d",
   "description":"Description of the new partition link.",
   "name":"Partition Link",
   "type":"smc-d"
}
```

Figure 391. Create Partition Link: Request

```
202
Server: Hardware management console API web server / 2.0
Location: /api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16
Cache-control: no-cache
Date: Tue, 29 Mar 2022 10:29:40 GMT
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/2436dc62-af4b-11ec-8dcd-fa163ee8cb0b"
}
```

Figure 392. Create Partition Link: Response

Delete Partition Link

The `Delete Partition Link` operation deletes the identified partition link asynchronously.

HTTP method and URI

```
POST /api/partition-links/{partition-link-id}/operations/delete
```

In this request, the URI variable *{partition-link-id}* is the object ID of the Partition Link object to be deleted.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Field Name	Type	Rqd/Opt	Description
`force-detach`	Boolean	Optional	An indication of whether active partitions associated with the partition link whose **object-id** is *{partition-link-id}* are detached forcefully. If set to **true**, any active partitions are forcefully detached before deleting the partition link. If set to **false** and one or more active partitions are associated with the partition link, the operation fails with status code 409 (Conflict) and reason code 100. Default: **false**

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Table 362. job-accepted-response nested object

Field Name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates of the asynchronous operation on the Partition link.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for "Query Job Status" on page 151. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason code" on page 758. The **job-results** field contains **null** when this operation is successful. When it is not successful, the **job-results** field contains an object with the following fields:

Table 363. Table error-job-results nested object

Field Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation asynchronously deletes the designated partition link. Once the delete request is accepted, a 202 (Accepted) status code is returned and the response body contains a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason code" on page 758.

The deletion first detaches all partitions currently associated with the partition link. The partition link stays in **"updating"** state until all partitions have been detached successfully. Only after detaching all partitions, the partition link itself is deleted. If the detach operation for a any partition fails due to an error, the deletion of the partition link is aborted and a job status and reason code is returned describing the failure. If the API user does not have action/task permission to the **Delete Partition Link** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the **object-id** *{partition-link-id}* does not identify a Partition Link object for which the API user has object-access permission. If the status of the partition link, the parent CPC, or any associated partition is not in a valid state, a 409

(Conflict) status code is returned. A 409 (Conflict) status code is also returned if the API user does not have object-access permission to all partitions currently associated with the partition link.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the partition link whose **object-id** is *{partition-link-id}*.
- Object-access permissions to all Partition objects currently connected to the partition link whose **object-id** is *{partition-link-id}*.
- Action/task permission to the **Delete Partition Link** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and the response body is provided as described in "Response body contents" on page 756.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code and associated error message.

Table 364. Delete Partition Link: HTTP status and reason codes

HTTP error status code	Reason code	Description
403 (Forbidden)	1	API user does not have the required action/task permissions.
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.

Table 364. Delete Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The operation cannot be performed because the partition link designated by the URI does not have a valid state. The valid states are **"complete"** and **"incomplete"**.
	6	The operation cannot be performed because the state of the CPC hosting the partition link is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	8	The operation cannot be performed because the request would result in the object being placed into a state that is inconsistent with its data model or other requirements. The request body contains a field whose presence or value is inconsistent with the current state of the object or some aspect of the system, and thus a prerequisite condition or dependency would no longer be met. For partition links of **type "smc-d"** or **"hipersockets"**: • The partition must have a minimum of one NIC to be associated with the partition link. It is not possible to remove the last NIC object of the partition. For partition links of **type "hipersockets"**: • The request cannot be processed because it would delete a NIC which is set as a **boot-network-device** of a partition.
	100	One or more partitions attached to the partition link designated by the *{partition-link-id}* parameter in the URI is in an active state. Either wait until all attached partitions go into a stopped state or retry the request with the **force-detach** field set to **true**.
	101	The user does not have permission to one or more partitions attached to the partition link designated by the *{partition-link-id}* parameter in the URI.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason code

HTTP error status code	Reason code	Description
204 (No Content)	N/A	The partition link was deleted successfully.

HTTP error status code	Reason code	Description
409 (Conflict)	1	The operation cannot be performed because the partition link designated by the URI does not have a valid status. The valid states are **"complete"** and **"incomplete"**.
	6	The operation cannot be performed because the state of the CPC hosting the partition link is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	100	One or more partitions attached to the partition link designated by the *{partition-link-id}* parameter in the URI is in an active state. Either wait until all attached partitions go into a stopped state or retry the request with the **force-detach** field set to **true**.
	102	One or more of the partitions attached to the partition link is busy with other operations.
500 (Server Error)	100	The job failed due to an internal error. The message text describes the error reason.
	101	Partition link delete job timed out.

Example HTTP interaction

```
POST /api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16/operations/delete HTTP/1.1
x-api-session: 61cij6i72fadwx6xnr0tkf5tyr42yt9wke9173vgkjmwsnyr85
Content-Type: application/json
Content-Length: 23
{
   "force-detach":false
}
```

Figure 393. Delete Partition Link: Request

```
202
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 29 Mar 2022 10:29:50 GMT
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/2a293340-af4b-11ec-9c36-fa163ee8cb0b"
}
```

Figure 394. Delete Partition Link: Response

Get Partition Link Properties

The `Get Partition Link Properties` operation returns the complete set of properties defined for a single partition link.

HTTP method and URI

```
GET /api/partition-links/{partition-link-id}
```

In this request, the URI variable *{partition-link-id}* is the object ID of the target Partition Link object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Partition Link object as defined in the "Data model" on page 736. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

The `Get Partition Link Properties` operation returns the current values of the properties for the Partition Link object as defined in the "Data model" on page 736.

The URI path must designate an existing Partition Link object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined in the "Data model" on page 736

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the partition link whose **object-id** is *{partition-link-id}*.

HTTP status and reason codes

On success, the HTTP status code 200 (OK) is returned and the response body is provided as described in the "Response body contents" on page 760.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 365. Get Partition Link Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16 HTTP/1.1
x-api-session: 61cij6i72fadwx6xnr0tkf5tyr42yt9wke9173vgkjmwsnyr85
```

Figure 395. Get Partition Link Properties: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 29 Mar 2022 10:29:41 GMT
Content-Type: application/json
Content-Length: 1065
{
   "adapter-uri":"/api/adapters/22856f78-af4b-11ec-ae6a-fa163eaa0f16",
   "bus-connections":[
      {
         "nics":[
            {
               "device-numbers":[
                  "0001"
               ],
               "fid":4096,
               "uuid":"22e5a410-af4b-11ec-ae6a-fa163eaa0f16"
            },
            {
               "device-numbers":[
                  "0002"
               ],
               "fid":4097,
               "uuid":"22e5a87a-af4b-11ec-ae6a-fa163eaa0f16"
            }
         ],
         "partition-name":"Partition 1",
         "partition-uri":"/api/partitions/51b79be4-ac13-11ec-86a7-fa163eaa0f16"
      },
      {
         "nics":[
            {
               "device-numbers":[
                  "0001"
               ],
               "fid":4098,
               "uuid":"22e5be3c-af4b-11ec-ae6a-fa163eaa0f16"
            },
            {
               "device-numbers":[
                  "0002"
               ],
               "fid":4099,
               "uuid":"22e5c396-af4b-11ec-ae6a-fa163eaa0f16"
            }
         ],
         "partition-name":"Partition 2",
         "partition-uri":"/api/partitions/9a417984-ac13-11ec-b798-fa163eaa0f16"
      }
   ],
   "class":"partition-link",
   "cpc-name":"4X142543",
   "cpc-uri":"/api/cpcs/a4aa2e69-9205-38ae-b55e-4d6ada6dea6d",
   "description":"Description of the new partition link.",
   "name":"Partition Link",
   "object-id":"22e58340-af4b-11ec-ae6a-fa163eaa0f16",
   "object-uri":"/api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16",
   "parent":"/api/console",
   "pending-operations":[],
   "starting-fid":4096,
   "state":"complete",
   "type":"smc-d"
}
```

Figure 396. Get Partition Link Properties: Response

List Partition Links

The `List Partition Links` operation lists the partition links known to the target Console.

HTTP method and URI

```
GET /api/partition-links
```

Query parameters:

Name	Type	Rqd/Opt	Description
`cpc-uri`	String/ URI	Optional	Filter string to limit returned objects to those that have a matching **cpc-uri** property.
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`state`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **state property**. Value must be a valid partition link state property value.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties **cpc-uri**, **state**, **name**, **type**, **object-uri**. This is a list of comma-separated strings where each string is a property name defined in the Partition Link object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`partition-links`	Array of partition -link- info objects	Array of partition-link-info objects, where each partition-link-info object is a JSON object with properties specified using the additional properties query parameter in addition to **cpc-uri, name, state, type, object-uri** which will be returned by default.

Each nested partition-link-info object contains the following fields:

Field Name	Type	Description
`cpc-uri`	String/ URI	The **cpc-uri** property of the Partition Link object.
`state`	String Enum	The **state** property of the Partition Link object.
`name`	String (1-64)	The **name** property of the Partition Link object.
`type`	String Enum	The **type** property of the Partition Link object.
`object-uri`	String/ URI	The **object-uri** property of the Partition Link object.

Description

This operation lists the partition links that are known to the target Console. The object URI, name, state, type and the URIs of its associated CPC are provided for each by default. Additional properties defined in the Partition Link object data model can be included by listing them in the **additional-properties** query parameter.

If the **name** query parameter is specified, the returned list is limited to those partition links that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the **state** query parameter is specified, the parameter is validated to ensure it is a valid value for the partition link **state** property according to the data model. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those partition links that have a **state** property matching the specified value. If the **state** parameter is omitted, this filtering is not done.

If the **cpc-uri** query parameter is specified, the returned list is limited to those partition links that have a matching **cpc-uri** property. If the **cpc-uri** parameter is omitted, this filtering is not done.

A partition link is included in the list only if the API user has object-access permission for that object. If the API user does not have permission to a partition link, that object is simply omitted from the list but no error status code results.

If no partition links are to be included in the results due to filtering or lack of object-access permission, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the Partition Link objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 762.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and any associated error message.

Table 366. List Partition Links: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/partition-links HTTP/1.1
x-api-session: 61cij6i72fadwx6xnr0tkf5tyr42yt9wke9173vgkjmwsnyr85
```

Figure 397. List Partition Links: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 29 Mar 2022 10:29:40 GMT
Content-Type: application/json
Content-Length: 213
{
   "partition-links":[
      {
         "cpc-uri":"/api/cpcs/a4aa2e69-9205-38ae-b55e-4d6ada6dea6d",
         "name":"Partition Link",
         "object-uri":"/api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16",
         "state":"complete",
         "type":"smc-d"
      }
   ]
}
```

Figure 398. List Partition Links: Response

Modify Partition Link

The `Modify Partition Link` operation modifies the identified partition link asynchronously.

HTTP method and URI

```
POST /api/partition-links/{partition-link-id}/operations/modify
```

In this request, the URI variable *{partition link-id}* is the object ID of the Partition Link object being modified.

Request body contents

The request body is a JSON object with the following fields:

Field Name	Type	Rqd/Opt	Description	Supported "type" values
`name`	String (1-64)	Optional	The value to be set as the partition link's **name** property.	All
`description`	String (0-200)	Optional	The value to be set as the partition link's **description** property.	All
`removed-partition-uris`	Array of String/ URI	Optional	The list of URIs of Partition objects to be removed from the partition link.	All
`added-connections`	Array of new-bus-connection objects	Optional	A list of bus connection elements to be added to the partition link.	smc-d, hipersockets
`modified-connections`	Array of modified-bus-connection objects	Optional	A list of modified bus connection elements. Possible modifications are: • Adding additional NICs to an existing bus connection • Removing NICs from an existing bus connection • Updating properties of existing NICs (e.g. device numbers)	smc-d, hipersockets

Field Name	Type	Rqd/Opt	Description	Supported "type" values
`starting-fid`	Integer	Optional	Starting number for the Functional ID assignment. If no FID for a NIC is provided, the next available FID starting from **fid-schema** will be used. Only valid when **type** is **smc-d**.	smc-d
`starting-device-number`	String (4)	Optional	The value to be set as the partition link's **starting-device-number** property. Only valid for link **type "hipersockets"**. The valid range is 0000-FFFF.	hipersockets
`maximum-transmission-unit-size`	Integer	Optional	The value to be set as the partition link's **maximum-transmission-unit-size** property. Only valid for the link **type "hipersockets"**. Constraint: This property can only be updated when the **status** of all attached partitions is **"stopped"** or **"reservation-error"**. Default: 8	hipersockets
`devices-per-path`	Integer (1-16)	Optional	Represents the number of devices present in each associated partition for each path in the partition link. • On incrementing the count, the difference in devices will be added to each associated partition for each path in the partition link. • On decrementing the count, the difference in devices will be removed from each associated partition for each path in the partition link. • If not provided, then no change occurs in the number of devices present in each associated partition in the partition link.	ctc
`added-partition-uris`	Array of String/ URI	Optional	The list of canonical URIs of Partition objects to be added to the partition link.	ctc
`removed-paths`	Array of removed-ctc-path-info objects	Optional	An array of one or more paths represented by removed-ctc-path-info nested objects (as described in Table 369 on page 767) where each element defines a physical path that is to be removed from existing paths used fro communication between partitions of the partition link. All the devices owned by all the partitions associated with the provided path will be removed from the partition link and the provided path will no longer be associated to the partition link.	ctc
`added-paths`	Array of added-ctc-path-info objects	Optional	An array of one or more paths represented by added-ctc-path-info nested objects (described in Table 349 on page 745) where each element defines a new physical path that is used for communicating between the partitions associated with the partition link.	ctc
`modified-paths`	Array of modified-ctc-path-info nested objects	Optional	An array of one or more paths represented by modified-ctc-path-info nested objects (as described in Table 368 on page 766), where each element represents an existing physical path of the partition link that has to be modified. This property is used for modifying devices of an existing path and to modify an existing physical path of the partition link..	ctc

Each nested modified-bus-connection object contains the following fields:

Table 367. modified-bus-connection nested object				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`partition-uri`	String/ URI	Required	The canonical URI path of the partition to which this bus connection attaches.	smc-d, hipersockets
`removed-nics-uuids`	Array of String (36)	Optional	A list of UUIDs representing NIC elements removed from the specified partition.	smc-d, hipersockets
`number-of-added-nics`	Integer	Optional	Specifies how many NICs should be created. Additional details like **device-number** or FID can be specified with the **added-nics** field. When **type** is **"smc-d"**, valid values are 1-255. When **type** is **"hipersockets"**, valid values are 1-4096.	smc-d, hipersockets
`modified-nics`	Array of modified-nic objects	Optional	A list of modified NIC elements owned by the specified partition.	smc-d, hipersockets
`added-nics`	Array of new-nic objects	Optional	A list of NIC elements that will be owned by the specified partition.	smc-d, hipersockets

Each nested modified-nic object contains the following fields:

Table 368. modified-nic nested object				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`device-numbers`	Array of String (4)	Optional	The value to be set as the NIC's **device-numbers** property. Each device number is a string in the form of a 4-digit hexadecimal number. If the **type** of the partition link is **"hipersockets"**, a set of up to 3 consecutive device numbers can be specified and the allowed value range is 0000-FFFF. If the **type** of the partition link is **"smc-d"**, only one device number is allowed and the value range is 0001-FFFF. Default: auto-generated Constraint: If the **type** of the partition link is **"smc-d"**, this number must be unique across the device numbers of all other NIC elements of **type "roce"** or **"cna"** and all instances of the objects listed in "PCI-based device numbers" on page 197 associated with the partition. If the **type** is **"hipersockets"**, the number(s) must be unique across the device numbers of all other partition links of **type "hipersockets"** and all instances of objects listed in "Channel-based device numbers" on page 197 of the partition.	smc-d, hipersockets
`fid`	Integer	Optional	Functional ID of the NIC. Only valid when **type** is **"smc-d"**. The FID must be unique across all FIDs defined on the hosting CPC. Default: auto-generated Constraint: The value must be within the range defined by the values of the **minimum-fid-number** and **maximum-fid-number** properties of the hosting CPC.	smc-d

Table 368. modified-nic nested object (continued)				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`uuid`	String (36)	Required	UUID representing the NIC element object owned by the partition.	smc-d, hipersockets
`vlan-id`	Integer (1-4094)	Optional	The VLAN ID associated wit this NIC. This value can only be set when the **type** of the link is **"hipersockets"**.	hipersockets
`mac-address`	String	Optional	The MAC address associated with this NIC. It must be unique among all the NICs created in the CPC. The MAC address is represented as six groups of two lower-case hexadecimal digits separated by colons (:). Only locally administered unicast MAC addresses are valid, for example, "02:ff:12:34:56:78". This value can only be set when the **type** of the link is **"hipersockets"**. Default: auto-generated	hipersockets

Each nested removed-ctc-path-info object contains the following fields:

Table 369. removed-ctc-path-info nested object				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`adapter-port-uri`	String/ URI	Required	Canonical URI path of the FICON Adapter object port.	ctc
`connecting-adapter-port-uri`	String/URI	Required	Canonical URI path of the connecting FICON Adapter port.	ctc

Each nested modified-ctc-path-info object contains the following fields:

Table 370. modified-ctc-path-info nested object				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`starting-device-number`	String (4)	Optional	The value to be set as the path's **starting-device-number** property. The **starting-device-number** is a string in the form of a 4-digit hexadecimal number. The allowed range is 0000-FFFF. If not provided, the value will be automatically generated the default value is 4000.	ctc

Table 370. modified-ctc-path-info nested object (continued)				
Field Name	**Type**	**Rqd/Opt**	**Description**	**Supported "type" values**
`devices`	Array of ctc-endpoint objects	Optional	Each element specifies the devices owned by all associated partitions in the partition link. An array of ctc-endpoint objects, each ctc-endpoint has a pair of endpoints between two partitions to form a valid communication path. Adding a new path: • It is optional to provide **devices** for a new path. The devices will be automatically generated for all partitions associated with the partition link if no **devices** are provided in the request. • It is allowed to provided **devices** for some of the partition combinations and **devices** are automatically generated for the rest of the partitions associated with the partition link. The number of devices to be provided in the **device-numbers** must be equal to **devices-per-path** provided in the request The **endpoint-pair** array must always have 2 entries representing a connection between a partition pair. Constraint: **devices** is an optional field. If provided, the number of device numbers provided in **device-numbers** field must always be equal to the **devices-per-path** count.	ctc
`adapter-port-uri`	String/ URI	Required	Canonical URI path of the FICON Adapter object that represents the adapter port of the path.	ctc
`connecting-adapter-port-uri`	String/ URI	Required	Canonical URI path of the FICON Adapter object that represents the connecting adapter port of the path.	ctc
`new-adapter-port-uri`	String/ URI	Optional	Canonical URI of the FICON Adapter object representing a new adapter port to be used for an existing path.	ctc
`new-connecting-adapter-port-uri`	String/ URI	Optional	Canonical URI of the FICON Adapter object representing a new connecting adapter port to be used for an existing path.	ctc

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Table 371. job-accepted-response nested object		
Field Name	**Type**	**Description**
`job-uri`	String/ URI	URI that may be queried to retrieve status updates of the asynchronous operation on the Partition link.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the "Query Job Status" on page 151. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 777. When this operation is successful, the **job-results** field contains an object with the following fields:

Field Name	Type	Description
`operation-results`	Array of attach-and-detach-operation-status objects	Array of object in which each object represents the attachment or detachment status of user requested partitions.

Each nested attach-and-detach-operation-status object contains the following fields:

Field Name	Type	Description
`partition-uri`	String/ URI	The canonical URI path of the Partition object that is being added/removed.
`operation-status`	String Enum	The status of the asynchronous attach and detach operation result: • **"attached"** - The partition is successfully attached. • **"detached"** - The partition is successfully detached. • **"pending-retry"** - The attach or detach operation failed in the first attempt. The operation will be retried with the next SE restart. • **"failed"** - The attach or detach operation failed and the requested operation is canceled. One possible scenario could be that the requested partition no longer exists on the system.

When it is not successful, the **job-results** field contains an object with the following fields:

Table 372. error-job-results nested object

Field Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

This operation asynchronously updates one or more of the writable properties of the designated partition link with the values specified. Once the modify request is accepted, a 202 (Accepted) status code is returned and the response body contains a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 777.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

The operation always returns a job-URI even if the request does not contain any bus connections to be added or removed. In this case the job will complete immediately.

The designated partition link will turn into **"updating"** state until all requested partition attach or detach operations have completed successfully.

If the API user does not have action/task permission to the **Partition Link Details** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if the object-id *{partition-link-id}* does not identify a Partition Link object for which the API user has object-access permission. If the status of the partition link, the parent CPC, or any associated partition is not in a valid state, a 409 (Conflict) status code is returned. A 409 (Conflict) status code is also returned if the API user does not have object-access permission to all partitions currently associated with the partition link.

SMC-D and HiperSockets Partition Links

If asynchronous attach or detach operation fails due to an error, the partition link will stay in **"updating"** state and the operation will be retried after an SE reboot. If retries for an attach operation are not successful, this partition will be removed from the partition link. If retries for a detach operation are not successful, the partition will remain in the partition link. Both kinds of failure lead to the creation of a hardware message. The asynchronous job completes after all partition attach or detach operations have either succeeded or failed once. The job-results property contains further details on the outcome of the individual attachments or detachment of partitions.

CTC Partition Links

The connectivity between the partitions can be incremented by adding additional paths represented with **added-paths** in the request. Each entry in the **added-paths** array represents a new CTC path to be defined through which the connections are formed between the partitions provided in the request. A valid CTC path can be defined by providing **adapter-port-uri** and **connecting-adapter-port-uri** representing FICON adapters enabled in CTC mode. The defined path will be known as: **switched** if both the provided adapters are connected to same switch, **switched-loopback** if both the provided adapters are representing same FICON Adapter, and **point-to-point** if both the provided adapters are connected to each other.

Each path can optionally have **devices** array representing the devices to be created on each partition provided in the request. Each entry in the **devices** array represent an **endpoint-pair**. It is optional to provide **endpoint-pair** in the request. If provided the maximum number of **endpoint-pair** entries must be less than or equal to the total number of combinations possible between the partitions provided in **added-partitions** field in the request and existing partitions associated with the partition link. If **endpoint-pair** entries are not provided the devices will be automatically generated based on the **starting-device-number** provided in the request. The **starting-device-number** is automatically generated if not provided in the request. Example: If 2 partitions are provided in **added-partition-uris** field with 2 existing partitions associated with the partition link, the number of **endpoint-pair** entries must be 6. Existing partitions in the partition link = {A, B}, Added partitions = {C,D}, endpoint pairs = {(A,B),(A,C),(A,D),(B,C),(B,D),(C,D)}.

The connections between the partitions can be altered by removing existing paths represented with **removed-paths** in the request. Each entry represent an existing path identified by **adapter-port-uri** and **connecting-adapter-port-uri**. All the connections represented by the **removed-paths** will be removed for all the partitions associated with the partition link. All existing paths associated with the partition link can be altered by providing **modified-paths** array in the request. The source or target adapter of an existing path can be altered by providing **new-adapter-port-uri** or **new-connecting-adapter-port-uri**. The optional devices array represents the devices to be created on the altered path. If no **devices** are provided in the request, all the connections represented by **devices** in the existing path will be used in the altered path. The **devices** in the **modified-paths** field represents the modified device numbers of an existing path. Each entry in the **devices** array represents **endpoint-pair** array and each entry in the **endpoint-pair** must have 2 entries representing a connection between an existing partition pair. The **device-numbers** in each **endpoint-pair** entry represents an array of all modified and unmodified devices of an existing partition. The number of **devices** in the **device-numbers** must be equal to the **devices-per-path** in the request.

If the modify operation fails due to an error, then it will undo all the changes that the user requested as part of the modify request. None of the failed operations gets retried again and the state of the partition link changes from **"updating"** to its previous state before the modification.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the partition link whose **object-id** is *{partition-link-id}*.
- Object-access permission to all Partition objects designated by a **partition-uri** field.
- Object-access permission to all FICON Adapter objects designated by **adapter-port-uri**, **connecting-adapter-port-uri**, **new-adapter-port-uri**, and **new-connecting-adapter-port-uri** fields.
- Action/task permission to the **Partition Link Details** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in “Response body contents” on page 768 .

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

Table 373. Modify Partition Link: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The value of a field does not provide a unique value for the corresponding data model property as required. The partition link **name** provided by the user is already in use by another partition link associated with the hosting CPC. For partition links of **type "ctc"**, the **partition-uri** combination in the **endpont-pair** in the request must be unique.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body. A prerequisite condition or dependence among request body fields is not met. For partition links of **type "ctc"**: • The requested number of devices in the **device-numbers** must be equal to the **devices-per-path** of the partition link. • The newly added **partition-uri** provided in the **endpoint-pair** of the request must be provided in **partition-uris** for a create request and **added-partition-uris** for a modify request. • The **partition-uri** provided in **removed-partition-uris** must not be provided in any **endpoint-pair** in the request. • The same **partition-uri** cannot be provided in **added-partition-uris** and **removed-partition-uris**. • The same path cannot be provided in both **paths** and **removed-paths**.
	452	For partition links of **type "smc-d"**: • The provided **device-numbers** of any of the NICs is already in use by an instance of one of the objects listed in "PCI-based device numbers" on page 197 of the partition. For partition links of **type "hipersockets"**: • The provided **device-numbers** of any of the NICs is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197 of the partition. For partition links of **type "ctc"**: • The provided **device-numbers** of any **ctc-partition-devices-endpoint** is already in use by an instance of one of the objects listed in "Channel-based device numbers" on page 197 of the partition. The **error-details** field of the response body contains a conflicting-device-numbers object (as described in Table 354 on page 753) listing the device number values that are not unique.

Table 373. Modify Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	455	For partition links of **type "smc-d"**: • The provided **fid** of any of the NICs is already in use on the hosting CPC. The **error-details** field of the response body contains a invalid-fid-details object listing the FID values that are not unique.
	456	For partition links of **type "ctc"**, the provided combination of **adapter-uris** in **paths** does not form a valid path.
	458	For partition links of **type "ctc"**, one of the adapters provided in the **paths** is not configured in FICON CTC mode.
	460	For partition links of **type "hipersockets"**, the provided **macs** of any of the NICs is already in use on the hosting CPC. The **error-details field** of the response body contains an **invalid-mac-details** object listing the MAC values that are not unique.
	462	For partition links of **type "ctc"**, the FICON adapter URI provided to define a path is already used by another path defined in the request.

Table 373. Modify Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	API user does not have the required action/task permissions.
	450	For partition links of **type "smc-d"**: • The value of **starting-fid** of the partition link cannot be altered as the API user does not have object-access permission to one or more partitions associated with the partition link. For partition links of **type "hipersockets"**: • The value of **starting-device-number** of the partition link cannot be altered as the API user does not have object-access permission to one or more partitions associated with the partition link. For partition links of **type "ctc"**, The following operations are not allowed as API user does not have object-access permission to one or more partitions associated with the partition link: • Editing the **devices-per-path** of the partition link • Editing the **starting-device-number** of the path in the partition link • Editing the **adapter-port-uri** and **connecting-adapter-port-uri** of the path in the partition link • Adding new paths to the partition link • Removal of an existing path from the partition link • Adding new partitions to the partition link.
	451	For partition links of **type "ctc"**, the following operations are not allowed as the API user does not have object-access permission to one or more FICON adapters used in defining the paths of the partition link. • Editing the **devices-per-path** of the partition link • Adding new paths to the partition link. • Adding new partitions to the partition link.
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.
	2	A URI in the request body does not designate an existing resource of the expected type or designates a resource for which the user does not have object-access permission. The **error-details** field of the response body contains a nic-not-found-details object defined in Table 374 on page 777, which identifies the objects that could not be found.
	101	One or more NIC UUIDs in the request body does not designate an existing resource of the expected type, or the resources are no longer exist. The **error-details** field of the response body contains a nic-not-found-details object identifying the objects that could not be found.

Table 373. Modify Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	1	The operation cannot be performed because the partition link designated by the URI does not have a valid state. The valid states are **"complete"** and **"incomplete"**.
	2	The operation cannot be performed because the resource designated by the request URI is currently busy performing some other operation.
	6	The operation cannot be performed because the state of the CPC hosting the partition link is not valid to perform the operation. It must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, and **"exceptions"**.
	8	The operation cannot be performed because the request would result in the object being placed into a state that is inconsistent with its data model or other requirements. The request body contains a field whose presence or value is inconsistent with the current state of the object or some aspect of the system, and thus a prerequisite condition or dependency would no longer be met. For partition links of **type "smc-d"** or **"hipersockets"**: • The partition must have a minimum of one NIC to be associated with the partition link. It is not possible to remove the last NIC object of the partition. For partition links of **type "hipersockets"**: • The request cannot be processed because it would delete a NIC which is set as a **boot-network-device** of a partition.
	10	The operation cannot be performed because the affected SE is in the process of being shut down.
	100	The operation cannot be performed because the status of one or more attached partitions is not valid (must be **"stopped"** or **"reservation-error"**).
	102	The operation cannot be performed because one or more partitions provided in the request body are currently busy performing some other operations.
	117	One or more modified or removed **partition-uris** in the request body does not designate a partition that is associated with the partition link.
	550	For partition links of **type "smc-d"**, the CPC identified by the **cpc-uri** already has the maximum number of FIDs defined.
	551	Unable to generate new device numbers as there are not enough free device numbers available.
	552	For partition links of **type "smc-d"**, the provided FID of any of the NICs has reached the maximum allowed value on the CPC. The **error-details** field of the response body contains an invalid-fid-details object listing all the FIDs that are exceeding the maximum allowed value on the CPC.

Table 373. Modify Partition Link: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
	553	For partition links of **type "smc-d"**, the starting FID value is beyond the allowed FID value range in the system.
	554	The maximum number of allowed partitions for a partition link has been reached.
	555	For partition links of **type "smc-d"** or **"hipersockets"**: • One or more partitions provided in **added-connections** array is already part of the partition link. The **error-details** field of the response body contains a partitions-exist-details object, described in Table 375 on page 777 , identifying the partitions that are part of the partition link. For partition links of **type "ctc"**: • One or more partitions provided in **added-partition-uris** array is already part of the partition link. The **error-details** field of the response body contains a partitions-exist-details object, described in Table 376 on page 777 , identifying the partitions that are part of the partition link.
	556	For partition links of **type "smc-d"**, the maximum number of Function ID (FIDs) per Internal Shared Memory (ISM) adapter that may be created for the CPC has been exceeded.
	557	For partition links of **type "hipersockets"**, the operation failed because it requires the generation of one or more MAC addresses, but the range of available addresses has been exhausted. [Added by feature **dpm-hipersockets-partition-link-management**]
	558	For partition links of **type "ctc"**, the number of devices defined in the partition for a path has exceeded the limit of 256. [Added by feature **dpm-ctc-partition-link-management**]
	559	For partition links of **type "ctc"**, the **adapter-port-uri** and **connecting-adapter-port-uri** provided in the request does not designate any existing paths associated with the partition link. [Added by feature **dpm-ctc-partition-link-management**]
	561	For partition links of **type "hipersockets"**: • The maximum number of HiperSockets devices that may be created for the CPC has been exceeded. [Added by feature **dpm-hipersockets-partition-link-management**]
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

The nic-not-found-details object contains the following fields:

Table 374. nic-not-found-details nested object		
Field Name	**Type**	**Description**
`uuids`	Array of String (36)	List of UUIDs of NIC objects that are not found.

The partitions-exist-info object contains the following fields:

Table 375. partitions-exist-info nested object		
Field Name	**Type**	**Description**
`partitions-exist`	partitions-exist-details	List of all the partitions associated with the Partition object

The partitions-exist-details object contains the following fields:

Table 376. partitions-exist-details nested object		
Field Name	**Type**	**Description**
`partition-uris`	Array of String/URI	The value of this field contains all the partition URIs which are already part of the partition link.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Table 377. Modify Partition Link: Job status and reason codes		
HTTP error status code	**Reason code**	**Description**
200 (OK)	N/A	The Asynchronous job for modify partition link operation is completed. The detailed status of the operation is available in the **job-results** field in the response.
500 (Server Error)	100	The job failed due to an internal error. The message text describes the error reason.
	101	Partition link asynchronous attach/detach job timed out.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Example HTTP interaction

```
POST /api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16/operations/modify HTTP/1.1
x-api-session: 61cij6i72fadwx6xnr0tkf5tyr42yt9wke9173vgkjmwsnyr85
Content-Type: application/json
Content-Length: 647
{
   "added-connections":[
      {
         "nics":[
            {
               "device-numbers":[
                  "1a10"
               ],
               "fid":100
            }
         ],
         "number-of-nics":1,
         "partition-uri":"/api/partitions/7005db60-af47-11ec-9e5d-fa163eaa0f16"
      }
   ],
   "modified-connections":[
      {
         "added-nics":[
            {
               "device-numbers":[
                  "1a12"
               ],
               "fid":102
            }
         ],
         "modified-nics":[
            {
               "device-numbers":[
                  "1a11"
               ],
               "fid":101,
               "uuid":"22e5a410-af4b-11ec-ae6a-fa163eaa0f16"
            }
         ],
         "number-of-added-nics":2,
         "partition-uri":"/api/partitions/51b79be4-ac13-11ec-86a7-fa163eaa0f16",
         "removed-nics-uuids":[
            "22e5a87a-af4b-11ec-ae6a-fa163eaa0f16"
         ]
      }
   ],
   "name":"ModifiedPartitionLink",
   "removed-partition-uris":[
      "/api/partitions/9a417984-ac13-11ec-b798-fa163eaa0f16"
   ]
}
```

Figure 399. Modify Partition Link: Request

```
202
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 29 Mar 2022 10:29:44 GMT
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/26ca2c04-af4b-11ec-92d5-fa163ee8cb0b"
}
```

Figure 400. Modify Partition Link: Response

Inventory service data

Information about the Partition Links managed by the HMC can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Partition Link objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"partition-link"** are to be included. Information for a particular partition link is included only if the API user has object-access permission to that object.

For each partition link to be included, the inventory response includes an array entry for the Partition Link object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for "Get Partition Link Properties" on page 759. That is, the data provided is the same as would be provided if a `Get Partition Link Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a partition link. This object would appear as one array entry in the response array:

```
{
   "adapter-uri":"/api/adapters/22856f78-af4b-11ec-ae6a-fa163eaa0f16",
   "bus-connections":[
      {
         "nics":[
            {
               "device-numbers":[
                  "0001"
               ],
               "fid":4096,
               "uuid":"22e5a410-af4b-11ec-ae6a-fa163eaa0f16"
            },
            {
               "device-numbers":[
                  "0002"
               ],
               "fid":4097,
               "uuid":"22e5a87a-af4b-11ec-ae6a-fa163eaa0f16"
            }
         ],
         "partition-name":"Partition 1",
         "partition-uri":"/api/partitions/51b79be4-ac13-11ec-86a7-fa163eaa0f16"
      },
      {
         "nics":[
            {
               "device-numbers":[
                  "0001"
               ],
               "fid":4098,
               "uuid":"22e5be3c-af4b-11ec-ae6a-fa163eaa0f16"
            },
            {
               "device-numbers":[
                  "0002"
               ],
               "fid":4099,
               "uuid":"22e5c396-af4b-11ec-ae6a-fa163eaa0f16"
            }
         ],
         "partition-name":"Partition 2",
         "partition-uri":"/api/partitions/9a417984-ac13-11ec-b798-fa163eaa0f16"
      }
   ],
   "class":"partition-link",
   "cpc-name":"4X142543",
   "cpc-uri":"/api/cpcs/a4aa2e69-9205-38ae-b55e-4d6ada6dea6d",
   "description":"Description of the new partition link.",
   "name":"Partition Link",
   "object-id":"22e58340-af4b-11ec-ae6a-fa163eaa0f16",
   "object-uri":"/api/partition-links/22e58340-af4b-11ec-ae6a-fa163eaa0f16",
   "parent":"/api/console",
   "pending-operations":[],
   "starting-fid":4096,
   "state":"complete",
   "type":"smc-d"
}
```


Figure 401. Partition Link object: Sample inventory data - Response

Chapter 11. Core IBM zSystems resources

These APIs provide access to and control of the following IBM zSystems and LinuxONE HMC/SE objects:

- Console
- Group
- CPC
- Logical Partition

In addition, these APIs provide access to the following CPC-related data items:

- Reset Activation Profiles
- Image Activation Profiles
- Load Activation Profiles
- Group Profiles
- Capacity Records

Limited API support is available for Adapters attached to a CPC that is not enabled for DPM. The following Adapter operations, described in Chapter 10, "Dynamic Partition Manager (DPM)," on page 195, are supported:

- "List Permitted Adapters" on page 377
- "Update Adapter Firmware" on page 403

Operations Summary

Following are the operations summaries for each of the core IBM zSystems and LinuxONE objects.

Console operations summary

The following table provides an overview of the operations provided for Console objects.

Table 378. Core IBM zSystems resources - Console: operations summary

Operation name	HTTP method and URI path
"Get Console Properties" on page 812	`GET /api/console`
"Restart Console" on page 819	`POST /api/console/operations/restart`
"Shutdown Console" on page 821	`POST /api/console/operations/shutdown`
"Reorder User Patterns" on page 822	`POST /api/console/operations/reorder-user-patterns`
"Get Console Audit Log" on page 824	`GET /api/console/operations/get-audit-log`
"Get Console Security Log" on page 830	`GET /api/console/operations/get-security-log`
"Get Console Events Log" on page 833	`GET /api/console/operations/get-events-log`

© Copyright IBM Corp. 2022, 2024

Table 378. Core IBM zSystems resources - Console: operations summary (continued)	
Operation name	**HTTP method and URI path**
"List Console Hardware Messages" on page 839	`GET /api/console/hardware-messages`
"Get Console Hardware Message Properties" on page 841	`GET /api/console/hardware-messages/{hardware-message-id}`
"Delete Console Hardware Message" on page 843	`DELETE /api/console/hardware-messages/{hardware-message-id}`
"Request Console Service" on page 844	`POST /api/console/hardware-messages/{hardware-message-id}/operations/request-service`
"Get Console Service Request Information" on page 846	`GET /api/console/hardware-messages/{hardware-message-id}/operations/get-service-information`
"Decline Console Service" on page 848	`POST /api/console/hardware-messages/{hardware-message-id}/operations/decline-service`
"List Unmanaged CPCs" on page 850	`GET /api/console/operations/list-unmanaged-cpcs`
"Get Mobile App Preferences" on page 852	`GET /api/console/operations/get-mobile-app-preferences`
"Set Mobile App Preferences" on page 853	`POST /api/console/operations/set-mobile-app-preferences`
"Get CPC Notification Preferences for Device" on page 856	`POST /api/console/operations/get-device-cpc-notification-preferences`
"Update CPC Notification Preferences for Device" on page 860	`POST /api/console/operations/update-device-cpc-notification-preferences`
"List Remote Firmware Updates of the Console" on page 864	`GET /api/console/remote-firmware-updates`
"Get Console Remote Firmware Update Properties" on page 867	`GET /api/console/remote-firmware-updates/{remote-firmware-update-id}`
"Delete Console Remote Firmware Update" on page 868	`DELETE /api/console/remote-firmware-updates/{remote-firmware-update-id}`
"Authorize Remote Firmware Updates" on page 870	`POST /api/console/operations/authorize-remote-firmware-updates`

Table 378. Core IBM zSystems resources - Console: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Update Welcome Text" on page 871	`POST /api/console/operations/update-welcome-text`
"Get Console Notification Preferences for Device" on page 873	`POST /api/console/operations/get-device-console-notification-preferences`
"Update Console Notification Preferences for Device" on page 875	`POST /api/console/operations/update-device-console-notification-preferences`
"Console Single Step Install" on page 876	`POST /api/console/operations/single-step-install`
"Report a Console Problem" on page 882	`POST /api/console/operations/report-problem` [Added by feature **report-a-problem**]
"Console Delete Retrieved Internal Code" on page 884	`POST /api/console/operations/delete-retrieved-internal-code` [Added by feature **hmc-delete-retrieved-internal-code**]
"List Console API Features" on page 887	`GET /api/console/operations/list-features`

Table 379. Core IBM zSystems resources - Console: URI variables	
URI variable	**Description**
{hardware-message-id}	Element ID of the hardware message object
{remote-firmware-update-id}	Element ID of the Remote Firmware Update element object

User operations summary

The following table provides an overview of the operations provided for User objects.

Table 380. Core IBM zSystems resources - User: operations summary	
Operation name	**HTTP method and URI path**
"List Users" on page 900	`GET /api/console/users`
"Get User Properties" on page 902	`GET /api/users/{user-id}`
"Update User Properties" on page 904	`POST /api/users/{user-id}`
"Add User Role to User" on page 907	`POST /api/users/{user-id}/operations/add-user-role`

Table 380. Core IBM zSystems resources - User: operations summary (continued)	
Operation name	**HTTP method and URI path**
“Remove User Role from User” on page 909	`POST /api/users/{user-id}/operations/remove-user-role`
“Create User” on page 911	`POST /api/console/users`
“Delete User” on page 915	`DELETE /api/users/{user-id}`

Table 381. Core IBM zSystems resources - User: URI variables	
URI variable	**Description**
{user-id}	Object ID of the User object

User Role operations summary

The following table provides an overview of the operations provided for User Role objects.

Table 382. Core IBM zSystems resources - User Role: operations summary	
Operation name	**HTTP method and URI path**
“List User Roles” on page 921	`GET /api/console/user-roles`
“Get User Role Properties” on page 923	`GET /api/user-roles/{user-role-id}`
“Update User Role Properties” on page 925	`POST /api/user-roles/{user-role-id}`
“Add Permission to User Role” on page 927	`POST /api/user-roles/{user-role-id}/operations/add-permission`
“Remove Permission from User Role” on page 930	`POST /api/user-roles/{user-role-id}/operations/remove-permission`
“Create User Role” on page 933	`POST /api/console/user-roles`
“Delete User Role” on page 935	`DELETE /api/user-roles/{user-role-id}`

Table 383. Core IBM zSystems resources - User Role: URI variables	
URI variable	**Description**
{user-role-id}	Object ID of the User Role object

Task operations summary

The following table provides an overview of the operations provided for Task objects.

Table 384. Core IBM zSystems resources - Task: operations summary	
Operation name	**HTTP method and URI path**
"List Tasks" on page 938	`GET /api/console/tasks`
"Get Task Properties" on page 940	`GET /api/console/tasks/{task-id}`

Table 385. Core IBM zSystems resources - Task: URI variables	
URI variable	**Description**
{task-id}	Element ID of the Task object

User Pattern operations summary

The following table provides an overview of the operations provided for User Pattern objects.

Table 386. Core IBM zSystems resources - User Pattern: operations summary	
Operation name	**HTTP method and URI path**
"List User Patterns" on page 946	`GET /api/console/user-patterns`
"Get User Pattern Properties" on page 947	`GET /api/console/user-patterns/{user-pattern-id}`
"Update User Pattern Properties" on page 949	`POST /api/console/user-patterns/{user-pattern-id}`
"Create User Pattern" on page 952	`POST /api/console/user-patterns`
"Delete User Pattern" on page 955	`DELETE /api/console/user-patterns/{user-pattern-id}`

Table 387. Core IBM zSystems resources - User Pattern: URI variables	
URI variable	**Description**
{user-pattern-id}	Element ID of the User Pattern object

Password Rule operations summary

The following table provides an overview of the operations provided for Password Rule objects.

Table 388. Core IBM zSystems resources - Password Rule: operations summary	
Operation name	**HTTP method and URI path**
"List Password Rules" on page 961	`GET /api/console/password-rules`
"Get Password Rule Properties" on page 963	`GET /api/console/password-rules/{password-rule-id}`

Table 388. Core IBM zSystems resources - Password Rule: operations summary (continued)	
Operation name	**HTTP method and URI path**
“Update Password Rule Properties” on page 965	`POST /api/console/password-rules/{password-rule-id}`
“Create Password Rule” on page 967	`POST /api/console/password-rules`
“Delete Password Rule” on page 969	`DELETE /api/console/password-rules/{password-rule-id}`

Table 389. Core IBM zSystems resources - Password Rule: URI variables	
URI variable	**Description**
{password-rule-id}	Element ID of the Password Rule object

LDAP Server Definition operations summary

The following table provides an overview of the operations provided for LDAP Server Definition objects.

Table 390. Core IBM zSystems resources - LDAP Server Definition: operations summary	
Operation name	**HTTP method and URI path**
“List LDAP Server Definitions” on page 977	`GET /api/console/ldap-server-definitions`
“Get LDAP Server Definition Properties” on page 978	`GET /api/console/ldap-server-definitions/{ldap-server-definition-id}`
“Update LDAP Server Definition Properties” on page 980	`POST /api/console/ldap-server-definitions/{ldap-server-definition-id}`
“Create LDAP Server Definition” on page 982	`POST /api/console/ldap-server-definitions`
“Delete LDAP Server Definition” on page 984	`DELETE /api/console/ldap-server-definitions/{ldap-server-definition-id}`

Table 391. Core IBM zSystems resources - LDAP Server Definition: URI variables	
URI variable	**Description**
{ldap-server-definition-id}	Element ID of the LDAP Server Definition object

MFA Server Definition operations summary

The following table provides an overview of the operations provided for MFA Server Definition objects.

Table 392. Core IBM zSystems resources - MFA Server Definition: operations summary	
Operation name	**HTTP method and URI path**
“List MFA Server Definitions” on page 988	`GET /api/console/mfa-server-definitions`
“Get MFA Server Definition Properties” on page 990	`GET /api/console/mfa-server-definitions/{mfa-server-definition-id}`
“Update MFA Server Definition Properties” on page 991	`POST /api/console/mfa-server-definitions/{mfa-server-definition-id}`
“Create MFA Server Definition” on page 993	`POST /api/console/mfa-server-definitions`
“Delete MFA Server Definition” on page 995	`DELETE /api/console/mfa-server-definitions/{mfa-server-definition-id}`

Table 393. Core IBM zSystems resources - MFA Server Definition: URI variables	
URI variable	**Description**
{mfa-server-definition-id}	Element ID of the MFA Server Definition object

Group operations summary

The following table provides an overview of the operations provided for Group objects.

Table 394. Core IBM zSystems resources - Group: operations summary	
Operation name	**HTTP method and URI path**
“List Custom Groups” on page 998	`GET /api/groups`
“Get Custom Group Properties” on page 1000	`GET /api/groups/{group-id}`
“Create Custom Group” on page 1002	`POST /api/groups`
“Delete Custom Group” on page 1004	`DELETE /api/groups/{group-id}`
“List Custom Group Members” on page 1008	`GET /api/groups/{group-id}/members`
“Add Member to Custom Group” on page 1005	`POST /api/groups/{group-id}/operations/add-member`
“Remove Member from Custom Group” on page 1007	`POST /api/groups/{group-id}/operations/remove-member`

Table 395. Core IBM zSystems resources - Groups: URI variables	
URI variable	**Description**
{group-id}	Object ID of a Group object

CPC operations summary

The following tables provide an overview of the operations provided for CPC objects.

Table 396. Core IBM zSystems resources - CPC: operations summary	
Operation name	**HTTP method and URI path**
"List CPC Objects" on page 1036	`GET /api/cpcs`
"Get CPC Properties" on page 1039	`GET /api/cpcs/{cpc-id}`
"Update CPC Properties" on page 1048	`POST /api/cpcs/{cpc-id}`
"Start CPC" on page 1049	`POST /api/cpcs/{cpc-id}/operations/start`
"Stop CPC" on page 1052	`POST /api/cpcs/{cpc-id}/operations/stop`
"Activate CPC" on page 1054	`POST /api/cpcs/{cpc-id}/operations/activate`
"Deactivate CPC" on page 1057	`POST /api/cpcs/{cpc-id}/operations/deactivate`
"Import Profiles" on page 1059	`POST /api/cpcs/{cpc-id}/operations/import-profiles`
"Export Profiles" on page 1060	`POST /api/cpcs/{cpc-id}/operations/export-profiles`
"Set Auto-Start List" on page 1062	`POST /api/cpcs/{cpc-id}/operations/set-auto-start-list`
"Add Temporary Capacity" on page 1064	`POST /api/cpcs/{cpc-id}/operations/add-temp-capacity`
"Remove Temporary Capacity" on page 1067	`POST /api/cpcs/{cpc-id}/operations/remove-temp-capacity`
"Swap Current Time Server" on page 1069	`POST /api/cpcs/{cpc-id}/operations/swap-cts`
"Set STP Configuration" on page 1071	`POST /api/cpcs/{cpc-id}/operations/set-stp-config`
"Change STP-only Coordinated Timing Network" on page 1074	`POST /api/cpcs/{cpc-id}/operations/change-stponly-ctn`

Table 396. Core IBM zSystems resources - CPC: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Join STP-only Coordinated Timing Network" on page 1075	`POST /api/cpcs/{cpc-id}/operations/join-stponly-ctn`
"Leave STP-only Coordinated Timing Network" on page 1077	`POST /api/cpcs/{cpc-id}/operations/leave-stponly-ctn`
"Get CPC Audit Log" on page 1078	`GET /api/cpcs/{cpc-id}/operations/get-audit-log`
"Get CPC Security Log" on page 1081	`GET /api/cpcs/{cpc-id}/operations/get-security-log`
"Get CPC Events Log" on page 1084	`GET /api/cpcs/{cpc-id}/operations/get-events-log`
"List CPC Hardware Messages" on page 1088	`GET /api/cpcs/{cpc-id}/hardware-messages`
"Get CPC Hardware Message Properties" on page 1091	`GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}`
"Delete CPC Hardware Message" on page 1093	`DELETE /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}`
"Request CPC Service" on page 1095	`POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/request-service`
"Get CPC Service Request Information" on page 1097	`GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/get-service-information`
"Decline CPC Service" on page 1100	`POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/decline-service`
"Export WWPN List" on page 1101	`POST /api/cpcs/{cpc-id}/operations/export-port-names-list`
"Import DPM Configuration" on page 1104	`POST /api/cpcs/{cpc-id}/operations/import-dpm-config`
"List Remote Firmware Updates of a CPC" on page 1115	`GET /api/cpcs/{cpc-id}/remote-firmware-updates`
"Get CPC Remote Firmware Update Properties" on page 1117	`GET /api/ cpcs/{cpc-id} /remote-firmware-updates/{remote-firmware-update-id}`
"Delete CPC Remote Firmware Update" on page 1119	`DELETE /api/ cpcs/{cpc-id} /remote-firmware-updates/{remote-firmware-update-id}`

Table 396. Core IBM zSystems resources - CPC: operations summary (continued)	
Operation name	**HTTP method and URI path**
“Get Logical Partition Resource Assignments” on page 1121	`GET /api/cpcs/{cpc-id}/operations/get-lpar-resource-assignments`
“Get LPAR Controls” on page 1122	`GET /api/ cpcs/{cpc-id}/operations/get-lpar-controls`
“Update LPAR Controls” on page 1134	`POST /api/ cpcs/{cpc-id}/operations/update-lpar-controls`
“CPC Single Step Install” on page 1136	`POST /api/cpcs/{cpc-id}/operations/single-step-install`
“Import Secure Execution Key” on page 1142	`POST /api/cpcs/{cpc_id}/operations/import-se-key` [Added by feature **secure-execution-key-management**]
“Delete Secure Execution Key” on page 1145	`POST /api/cpcs/{cpc_id}/operations/delete-se-key` [Added by feature **secure-execution-key-management**]
“Import CPC Certificate” on page 1147	`POST /api/cpcs/{cpc-id}/operations/import-certificate` [Added by feature **secure-boot-with-certificates**]
“Report a CPC Problem” on page 1149	`POST /api/cpcs/{cpc-id}/operations/report-problem` [Added by feature **report-a-problem**]
“Get CPC Historical Sustainability Data” on page 1152	`POST /api/cpcs/{cpc-id}/operations/get-historical-sustainability-data` [Added by feature **environmental-metrics**]
“CPC Install and Activate” on page 1157	`POST /api/cpcs/{cpc-id}/operations/install-and-activate` [Added by feature **cpc-install-and-activate**]
“CPC Delete Retrieved Internal Code” on page 1162	`POST /api/cpcs/{cpc-id}/operations/delete-retrieved-internal-code` [Added by feature **cpc-delete-retrieved-internal-code**]
“List CPC API Features” on page 1165	`GET /api/cpcs/{cpc-id}/operations/list-features`
“Switch Support Elements” on page 1167	`POST /api/cpcs/{cpc-id}/operations/switch-support-elements` [Added by feature **switch-support-elements**]

Table 397. Core IBM zSystems resources - CPC: URI variables	
URI variable	**Description**
{cpc-id}	Object ID of a CPC object
{hardware-message-id}	Element ID of the hardware message object

Table 397. Core IBM zSystems resources - CPC: URI variables (continued)	
URI variable	**Description**
{remote-firmware-update-id}	Element ID of the Remote Firmware Update element object

Logical partition operations summary

The following tables provide an overview of the operations provided for Logical Partition objects.

Table 398. Core IBM zSystems resources - Logical partitions: operations summary	
Operation name	**HTTP method and URI path**
"List Logical Partitions of CPC" on page 1194	`GET /api/cpcs/{cpc-id}/logical-partitions`
"List Permitted Logical Partitions" on page 1196	`GET /api/console/operations/list-permitted-logical-partitions`
"Get Logical Partition Properties" on page 1200	`GET /api/logical-partitions/{logical-partition-id}`
"Update Logical Partition Properties" on page 1205	`POST /api/logical-partitions/{logical-partition-id}`
"Activate Logical Partition" on page 1207	`POST /api/logical-partitions/{logical-partition-id}/operations/activate`
"Deactivate Logical Partition" on page 1212	`POST /api/logical-partitions/{logical-partition-id}/operations/deactivate`
"Reset Normal" on page 1214	`POST /api/logical-partitions/{logical-partition-id}/operations/reset-normal`
"Reset Clear" on page 1216	`POST /api/logical-partitions/{logical-partition-id}/operations/reset-clear`
"Load" on page 1218	`POST /api/logical-partitions/{logical-partition-id}/operations/load-program` [Added by feature **secure-boot-with-certificates**]
"Load Logical Partition" on page 1224	`POST /api/logical-partitions/{logical-partition-id}/operations/load`
"Load Logical Partition from FTP" on page 1227	`POST /api/logical-partitions/{logical-partition-id}/operations/load-from-ftp`
"PSW Restart" on page 1230	`POST /api/logical-partitions/{logical-partition-id}/operations/psw-restart`
"Start Logical Partition" on page 1232	`POST /api/logical-partitions/{logical-partition-id}/operations/start`
"Stop Logical Partition" on page 1233	`POST /api/logical-partitions/{logical-partition-id}/operations/stop`
"Send OS Command" on page 1235	`POST /api/logical-partitions/{logical-partition-id}/operations/send-os-cmd`

Table 398. Core IBM zSystems resources - Logical partitions: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Open OS Message Channel" on page 1237	`POST /api/logical-partitions/{logical-partition-id}/operations/open-os-message-channel`
"List OS Messages of a Logical Partition" on page 1239	`GET /api/logical-partitions/{logical-partition-id}/operations/list-os-messages`
"Delete Logical Partition OS Message" on page 1243	`POST /api/logical-partitions/{logical-partition-id}/operations/delete-os-message`
"SCSI Load" on page 1245	`POST /api/logical-partitions/{logical-partition-id}/operations/scsi-load`
"SCSI Dump" on page 1248	`POST /api/logical-partitions/{logical-partition-id}/operations/scsi-dump`
"NVMe Load" on page 1251	`POST /api/logical-partitions/{logical-partition-id}/operations/nvme-load`
"NVMe Dump" on page 1254	`POST /api/logical-partitions/{logical-partition-id}/operations/nvme-dump`
"Report a Logical Partition Problem" on page 1260	`POST /api/logical-partitions/{logical-partition-id}/operations/report-problem` [Added by feature **report-a-problem**]
"Get Logical Partition Historical Sustainability Data" on page 1263	`POST /api/logical-partitions/{logical-partition-id}/operations/get-historical-sustainability-data` [Added by feature **environmental-metrics**]

Table 399. Core IBM zSystems resources - Logical partitions: URI variables	
URI variable	**Description**
{cpc-id}	Object ID of a CPC object
{logical-partition-id}	Object ID of a Logical Partition object

Certificate operations summary

The following tables provide an overview of the operations provided for Certificate objects. [Added by feature **secure-boot-with-certificates**]

Table 400. Core IBM zSystems resources - Certificate: operations summary	
Operation name	**HTTP method and URI path**
"Delete Certificate" on page 1267	`DELETE /api/certificates/{certificate-id}`
"Get Certificate Properties" on page 1268	`GET /api/certificates/{certificate-id}`
"Get Encoded Certificate" on page 1271	`GET /api/certificates/{certificate-id}/operations/get-encoded`

Table 400. Core IBM zSystems resources - Certificate: operations summary (continued)	
Operation name	**HTTP method and URI path**
"List Certificates" on page 1273	`GET /api/certificates`
"Update Certificate Properties" on page 1275	`POST /api/certificates/{certificate-id}`

Table 401. Core IBM zSystems resources - Certificate: URI variables	
URI variable	**Description**
{certificate-id}	Object ID of a Certificate object

Activation profile operations summary

The following tables provide an overview of the operations provided for the various types of Activation Profile objects.

Table 402. Core IBM zSystems resources - Reset activation profile: operations summary	
Operation name	**HTTP method and URI path**
"List Reset Activation Profiles" on page 1279	`GET /api/cpcs/{cpc-id}/reset-activation-profiles`
"Get Reset Activation Profile Properties" on page 1281	`GET /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}`
"Update Reset Activation Profile Properties" on page 1283	`POST /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}`
"Create Reset Activation Profile" on page 1285	`POST /api/cpcs/{cpc-id}/reset-activation-profiles` [Added by feature **create-delete-activation-profiles**]
"Delete Reset Activation Profile" on page 1289	`DELETE /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}` [Added by feature **create-delete-activation-profiles**]

Table 403. Core IBM zSystems resources - Image activation profile: operations summary	
Operation name	**HTTP method and URI path**
"List Image Activation Profiles" on page 1307	`GET /api/cpcs/{cpc-id}/image-activation-profiles`
"Get Image Activation Profile Properties" on page 1310	`GET /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}`
"Update Image Activation Profile Properties" on page 1315	`POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}`

Table 403. Core IBM zSystems resources - Image activation profile: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Assign Certificate to Image Activation Profile" on page 1316	`POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}/operations/assign-certificate` [Added by feature **secure-boot-with-certificates**]
"Unassign Certificate from Image Activation Profile" on page 1319	`POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}/operations/unassign-certificate` [Added by feature **secure-boot-with-certificates**]
"Create Image Activation Profile" on page 1321	`POST /api/cpcs/{cpc-id}/image-activation-profiles` [Added by feature **create-delete-activation-profiles**]
"Delete Image Activation Profile" on page 1338	`DELETE /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}` [Added by feature **create-delete-activation-profiles**]

Table 404. Core IBM zSystems resources - Load activation profile: operations summary	
Operation name	**HTTP method and URI path**
"List Load Activation Profiles" on page 1346	`GET /api/cpcs/{cpc-id}/load-activation-profiles`
"Get Load Activation Profile Properties" on page 1349	`GET /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}`
"Update Load Activation Profile Properties" on page 1351	`POST /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}`
"Create Load Activation Profile" on page 1353	`POST /api/cpcs/{cpc-id}/load-activation-profiles` [Added by feature **create-delete-activation-profiles**]
"Delete Load Activation Profile" on page 1360	`DELETE /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}` [Added by feature **create-delete-activation-profiles**]

Table 405. Core IBM zSystems resources - Group profile: operations summary	
Operation name	**HTTP method and URI path**
"List Group Profiles" on page 1364	`GET /api/cpcs/{cpc-id}/group-profiles`
"Get Group Profile Properties" on page 1366	`GET /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`
"Update Group Profile Properties" on page 1369	`POST /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`

Table 405. Core IBM zSystems resources - Group profile: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Create Group Profile" on page 1370	`POST /api/cpcs/{cpc-id}/group-profiles` [Added by feature **create-delete-activation-profiles**]
"Delete Group Profile" on page 1373	`DELETE /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}` [Added by feature **create-delete-activation-profiles**]

Table 406. Core IBM zSystems resources - Activation profile: URI variables	
URI variable	**Description**
{cpc-id}	Object ID of a CPC object
{group-profile-name}	Group profile name
{image-activation-profile-name}	Image activation profile name
{load-activation-profile-name}	Load activation profile name
{reset-activation-profile-name}	Reset activation profile name

Capacity record operations summary

The following tables provide an overview of the operations provided for Capacity Record objects.

Table 407. Core IBM zSystems resources - Capacity record: operations summary	
Operation name	**HTTP method and URI path**
"List Capacity Records" on page 1378	`GET /api/cpcs/{cpc-id}/capacity-records`
"Get Capacity Record Properties" on page 1380	`GET /api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}`

Table 408. Core IBM zSystems resources - Capacity record: URI variables	
URI variable	**Description**
{cpc-id}	Object ID of a CPC object
{capacity-record-id}	Capacity record identifier

Adapter operations summary

The following tables provide an overview of the operations provided for Adapter objects.

Table 409. Core IBM zSystems resources - Adapter: operations summary	
Operation name	**HTTP method and URI path**
"List Permitted Adapters" on page 377	`GET /api/console/operations/list-permitted-adapters`

Table 409. Core IBM zSystems resources - Adapter: operations summary (continued)	
Operation name	**HTTP method and URI path**
"Update Adapter Firmware" on page 403	`POST /api/adapters/{adapter-id}/operations/update-firmware`

Shared nested objects

Some of the Core API objects share common nested objects and are documented here for ease of reference.

Table 410. ec-mcl-description object		
Field name	**Type**	**Description**
actions	Array of action objects	An optional array of pending action objects. This field is not provided when the EC MCL information pertains to the HMC. When the EC MCL information pertains to an SE, this field is only provided when the HMC is communicating with the CPC's SE, and this field will be null if the information is currently unavailable or there are no pending actions.
ec	Array of ec objects	An optional array of EC objects. When the EC MCL information pertains to an SE, this field is only provided when the HMC is communicating with the CPC's SE, and this field will be null if the information is currently unavailable.
lic-control-level	String (1-4) or String Enum	An optional field containing the Licensed Internal Code control level or a string enumeration value indicating this information is not available. The possible string enumeration values are: • **"not-available"** - The LIC control level is not present on the hard disk drive (HDD) of the system. When the EC MCL information pertains to an SE, this information can only be retrieved when the HMC is communicating with the CPC's SE; therefore, this field will be null if the information could not be retrieved.
driver-level	String (1-4) or String Enum	An optional field containing the driver level of the installed code or a string enumeration value containing information about why this information is not available. The possible string enumeration values are: • **"not-available" -** The installed driver level is not present on the hard disk drive (HDD) of the system. • "**not-service-authority"** - the API user doesn't have service authority. • **"unable-to-determine"** - the attempt to obtain EC information from the CPC returned no data. This value is only applicable when the EC MCL information pertains to a CPC. When the EC MCL information pertains to a CPC, this information can only be retrieved when the HMC is communicating with the CPC's SE; therefore, this field will be null if the information could not be retrieved.

Table 410. ec-mcl-description object (continued)		
Field name	**Type**	**Description**
bundle-level	String (1-4) or String Enum	An optional field containing the bundle level of the installed code or a string enumeration value indicating this information is not available. The possible string enumeration values are: • **"not-available" -** the installed bundle level is not present on the hard disk drive (HDD) of the system. When the EC MCL information pertains to an CPC, this information can only be retrieved when the HMC is communicating with the CPC's SE; therefore, this field will be null if the information could not be retrieved.
arom-info	String Enum	An optional field indicating if the system was loaded with a CDU (Concurrent Driver Upgrade) EC AROM in the form of a string enumeration value. The possible string enumeration values are: • **"concurrent-engineering-changes-arom"**- the system was loaded with a CDU EC AROM. • **"engineering-changes-arom"** - the system was not loaded with a CDU EC AROM. When the EC MCL information pertains to a CPC, this value will also be returned if an error occurred when trying to determine if the system was loaded with a CDU EC AROM. • **"not-available"** - An error occurred when trying to determine if the target HMC was loaded with a CDU EC AROM. This value is only applicable when the EC MCL information pertains to the HMC. When the EC MCL information pertains to an CPC, this information can only be retrieved when the HMC is communicating with the CPC's SE; therefore, this field will be null if the information could not be retrieved.

Table 411. action object		
Field name	**Type**	**Description**
type	String Enum	One of: • **"channel-config"** - channels pending a config on/off • **"coupling-facility-reactivation"** - at least one coupling facility pending reactivation • **"power-on-reset-tracking"** - there is a need for a power-on-reset
activation	String Enum	One of: • **"current"** - the action is for the current activation • **"next"** - the action is for the next install and activation.
pending	Boolean	Is the action pending (true) or not pending (false)

Table 412. ec object		
Field name	**Type**	**Description**
number	String (1-6)	Engineering Change stream identifier.

Table 412. ec object (continued)		
Field name	**Type**	**Description**
part-number	String (1-8)	Engineering Change stream part number.
type	String (1-32)	Engineering Change stream name.
description	String (1-65)	Engineering Change stream descriptive text.
mcl	Aray of mcl objects	The list of MicroCode Levels for this Engineering Change.

Table 413. mcl object		
Field name	**Type**	**Description**
type	String Enum	One of: • **"retrieved"** - a retrieved or staged level • **"activated"** - an activated or applied level • **"accepted"** - a committed level • **"installable-concurrent"** - a non-disruptive apply-able level • **"removable-concurrent"** - a non-disruptive reject-able level.
level	String (1-3)	Microcode level.
last-update	Timestamp	Time stamp of the last update, in the number of milliseconds since midnight January 1, 1970 UTC. A null object is returned if no updates have occurred.

Table 414. stp-config object		
Field name	**Type**	**Description**
stp-id	String (0-8)	If in STP-only or Mixed CTN, the STP identifier. Otherwise, an empty string. Valid characters are 0-9, a-z, A-Z, underscore(_) and dash(-).
etr-id	Integer (0-31)	ETR Identifier, if in ETR mode. If not in ETR mode, a null object is returned.
preferred-time-server	stp-node-object	Describes the Preferred Timer Server. This property is optional, only returned on a Get request when the information is set.
backup-time-server	stp-node-object	Describes the Backup Timer Server. This property is optional, only returned on a Get request when the information is set.
arbiter	stp-node-object	Describes the arbiter of the CTN. This property is optional, only returned on a Get request when the information is set.
current-time-server	String Enum	Describes the CPC's role in the CTN. One of: • **"preferred"** - CPC is the Preferred Time Server • **"backup"** - CPC is the Backup Time Server.

This object is used to identify a CPC to the STP services. When used as input on the `Set STP Configuration` operation, if the **object-uri** field is not provided, all other fields are required with the exception of the **target-name** and **node-name** fields which are ignored for the set operation. If the **object-uri** field is provided, all other fields are optional. If all fields are provided, the **object-uri** field is ignored.

Table 415. stp-node object

Field name	Type	Description
object-uri	String URI	If the CPC is known to the HMC, contains the CPC's object-uri. Otherwise, contains a null object
type	String (0-6)	The CPC machine type, right justified and left padded with zeros or a empty string
model	String (0-3)	The CPC machine model or a empty string
manuf	String (0-3)	The CPC manufacturer or a empty string
po-manuf	String (0-2)	The CPC plant of manufacturer or a empty string
seq-num	String (0-12)	The CPC sequence number or a empty string
node-name	String (1-8)	The CPC name
target-name	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request and the **object-uri** is not **null**.

Table 416. psw-description object

Field name	Type	Description
psw	String	Program Status Word (PSW) information for a single processor.
cpid	String (2)	The hexadecimal processor identifier, right justified and left padded with zeros.

Table 417. zaware-network object

Field name	Type	Description
chpid	String (2)	The required network adapter channel path identifier, in hexadecimal characters 0-9,a-f,A-F. The format is two hexadecimal digits (00-FF).
ipaddr-type	String Enum	Indicates how this network adapter's IP address is obtained. One of the following values: • **"dhcp"** - obtains an IP address through DHCP • **"link-local"** - obtains a link-local IP address • **"static"** - uses the specified IP address information.
vlan-id	Integer (0-4094)	If this network adapter is attached to a Virtual LAN, this field contains the VLAN identifier. Otherwise, a null object indicating that this network adapter is not attached to a Virtual LAN.

Table 417. zaware-network object (continued)		
Field name	**Type**	**Description**
static-ip-info	network-ip-info object	When **ipaddr-type** is **"static"**, contains the static IP information. Otherwise, contains a null object indicating that a static IP address is not used.

Table 418. ssc-network object		
Field name	**Type**	**Description**
chpid	String (2)	The required network adapter channel path identifier, in hexadecimal characters 0-9,a-f,A-F. The format is two hexadecimal digits (00-FF).
port	Integer	The number of the port on the network adapter identified by **chpid**. When the associated Support Element is at Version 2.13.1 or earlier, this property is not returned on a Get operation, and it cannot be specified on an Update operation.
ipaddr-type	String Enum	Indicates how this network adapter's IP address is obtained. One of the following values: • **"dhcp"** - obtains an IP address through DHCP • **"link-local"** - obtains a link-local IP address • **"static"** - uses the IP address information specified in **static-ip-info**.
vlan-id	Integer (0-4094)	If this network adapter is attached to a Virtual LAN, this field contains the VLAN identifier. Otherwise, a null object indicating that this network adapter is not attached to a Virtual LAN.
static-ip-info	network-ip-info object	When **ipaddr-type** is **"static"**, contains the static IP information. Otherwise, contains a null object indicating that a static IP address is not used.

Table 419. ip-info object		
Field name	**Type**	**Description**
type	String Enum	The type of IP address being provided. One of the following values: • **"ipv4"** - an IPv4 address is provided • **"ipv6"** - an IPv6 address is provided.
ip-address	String/ IPV4 address or String/ IPV6 address	The IP address to be used. The format of the string (IPv4 or IPv6) must be as indicated by the **type** field.

Table 420. network-ip-info object		
Field name	**Type**	**Description**
type	String Enum	The type of IP address being provided. One of the following values: • **"ipv4"** - an IPv4 address is provided • **"ipv6"** - an IPv6 address is provided.

Table 420. network-ip-info object (continued)		
Field name	**Type**	**Description**
ip-address	String/ IPV4 address or String/ IPV6 address	The IP address to be used. The format of the string (IPv4 or IPv6) must be as indicated by the **type** field.
prefix	Integer	The number of leading bits of ip-address that represent the network prefix. • When **type** is **"ipv4"** - valid values are 0-32 • When **type** is **"ipv6"** - valid values are 0-128.

Table 421. absolute-capping object		
Field name	**Type**	**Description**
type	String Enum	The type of absolute capping. One of the following values: • **"none"** - no absolute capping • **"processors"** - processor type absolute capping.
value	Float	When **type** is **"none"**, value is not specified. When **type** is **"processors"**, value is in the range .01...255.00 in increments of .01 and is the limit of usage independent of priority.

Console object

The Console object represents the single IBM zSystems Hardware Management Console (HMC) application or for BCPii interface users the Support Element application. The Console object offers a heterogeneous set of services and capabilities, from basic Console control operations to general Console information.

Object-access to the single Console object is automatic for all authenticated HMC API users. For BCPii interface users a valid X-API-Target-Name request header must be specified.

Data model

This object includes the properties defined in the "Base managed object properties schema" on page 100, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

The following class-specific specializations apply to the other base managed object properties:

Table 422. Console object: base managed object properties specializations			
Name	**Qualifier**	**Type**	**Description of specialization**
object-uri	—	String/ URI	The canonical URI path of the Console object, of the form `/api/console`
parent	—	String/ URI	A Console object has no parent, so this property is always a null object.

Table 422. Console object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
class	—	String	The class of a Console object is **"console"**.
name	(ro)	String	The installation assigned name
description	(ro)	String	This property is not supported, and returned as null.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 423. Console object: class specific additional properties			
Name	**Qualifier**	**Type**	**Description**
version	—	String (1-8)	The version number for the Console object.
ec-mcl-description	—	ec-mcl-description object	A nested object that describes the EC (Engineering Change) and MCL (Microcode Level) for the Console. Refer to the description of the ec-mcl-description object for details.
network-info	—	network-info object	A nested object describing the network information for this Hardware Management Console.
machine-info	—	machine-info object	A nested object describing BIOS characteristics of the machine on which the Hardware Management Console is executing.
cpc-machine-info	—	machine-info object	A nested object describing the machine containing the CPC for which the console is a feature.
has-hardware-messages	(pc)	Boolean	The Console object has hardware messages (true), or does not have hardware messages (false).
hardware-messages	(c)(pc)	Array of hardware-message objects	The complete list of all Console hardware messages, each identified by its URI. This list corresponds to the list provided by the `List Console Hardware Messages` operation. If the console has no hardware messages, then an empty array is provided. The list of returned hardware messages can change as the result of the new messages being dynamically added or removed by the infrastructure or due to hardware messages being deleted through the `Delete Console Hardware Message` operation. **Note:** This property is not returned by the `Get Console Properties` operation, and only sessions with permission to the Hardware Messages task will receive a property-change notification for this property.
mobile-app-preferences	(p)(pc)	mobile-app-preferences object	The JSON nested object containing the mobile app preferences, as described in Table 435 on page 807. This property can be retrieved through the `Get Mobile App Preferences` operation, and modified through the `Set Mobile App Preferences` operation.

Table 423. Console object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
sna-name	—	String (1-17)	The fully qualified SNA name of the Console.
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
shutdown-in-process	—	Boolean	True indicates that a shutdown or restart of the console is in process. False indicates that the console is currently not being shut down or restarted.
shutdown-delay-allowed	—	Boolean	True indicates that applications can delay shutdowns or restarts of the console. False indicates that applications are not allowed to delay shutdowns or restarts of the console.
shutdown-delay-remaining	—	Integer	The number of seconds remaining for the currently outstanding shutdown delays for the console. A zero value indicates there are no outstanding shutdown delays in effect.
shutdown-delay-apps	—	Array of String	The names of the applications that are currently delaying the shutdown or restart of the console. An empty array indicates there are no applications delaying shutdowns or restarts.
shutdown-delay-disable-reasons	—	Array of String	The reasons shutdown delays are currently disabled. An empty array indicates that shutdown delays are not currently disabled.
hma-info	—	hma-info object	A nested object describing the Hardware Management Appliance (HMA) information for this Hardware Management Console (HMC), as described in Table 437 on page 809. This is a HMC-only property. If the HMC is not part of a HMA, this property is null.

Table 424. network-info object properties		
Name	**Type**	**Description**
this-hmc	Array of detailed-network-info objects	The collection of network information for the local Hardware Management Console. The number of objects returned is a function of the machine model and type on which the Hardware Management Console is executing. This information is available in the **machine-info** property. **Note:** This property is only returned when the console is associated with a Hardware Management Console.
primary-se	Array of detailed-network-info objects	The collection of network information for the local primary Support Element console. **Note:** This property is only returned when the console is associated with a Support Element console.

Table 424. network-info object properties (continued)		
Name	**Type**	**Description**
alternate-se	Array of detailed-network-info objects	The collection of network information for the alternate Support Element console. **Note:** This property is only returned when the console is associated with a Support Element console. [Added by feature **adapter-network-information**]

Table 425. detailed-network-info properties		
Name	**Type**	**Description**
hmc-name	String (1-16)	The Hardware Management Console name **Note:** This property is only returned when the console is associated with a Hardware Managment Console.
name	String (1-8)	The Support Element console name. **Note:** This property is only returned when the console is associated with a Support Element console.
interface-name	String	The network interface name
domain-name	String (1-255)	The domain name configured for this network interface
is-private	Boolean	Whether the interface is private (true) or public (false).
mac	String (1-12)	The MAC address of this network interface.
ipv4-address	Array of ipv4-info objects	A collection of nested objects which describe the IPv4 addresses for this network interface.
ipv6-address	Array of ipv6-info objects	A collection of nested objects which describe the IPv6 addresses for this network interface.

Table 426. ipv4-info properties			
Name	**Qualifier**	**Type**	**Description**
subnet-mask	(pc)	String (1-15)	The IP mask value
ip-address	(pc)	String IPV4 address	The IPv4 address

Table 427. ipv6-info properties			
Name	**Qualifier**	**Type**	**Description**
prefix-length	(pc)	Integer	The number of leading bits of the IPv6 address that represent the network prefix.

Table 427. ipv6-info properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ip-address	(pc)	String IPV6 address	The IPv6 address

Table 428. machine-info properties		
Name	**Type**	**Description**
machine-type	String (1-4)	The type of machine.
machine-model	String (1-3)	The model of the machine.
machine-serial	String (1-10)	The serial number of the machine.

Table 429. hardware-message object properties		
Name	**Type**	**Description**
element-uri	String/ URI	The canonical URI path of the Console hardware message. The URI is in the following form: `/api/console/hardware-messages/{hardware-message-id}`, where *{hardware-message-id}* is the value of the **element-id** property of the hardware message.
element-id	String (36)	The unique identifier for the hardware message. The **element-id** is in the form of a UUID.
parent	String/ URI	The parent of a console hardware message is the Console object. The **parent** value is the canonical URI path for the console.
class	String	The **class** of a hardware message object is **"hardware-message"**.
timestamp	Timestamp	The **timestamp** represents the date and time when the hardware message was created.
service-supported	Boolean	Indicates whether or not this hardware message represents a problem for which service may be requested. **True** is returned if the hardware message supports service, or **false** is returned if it does not.
text	String	The text of the hardware message.
details	Object	A hardware-message-details object if there are hardware message details, or null if there are no hardware message details.

Every hardware-message-details object contains the following base properties:

Table 430. hardware-message-details base properties		
Name	**Type**	**Description**
type	String Enum	The type of detail available for this hardware message. Valid values: • **"basic"** - basic details in the form of one or more text messages. • **"common-problem"** - specific details for this problem-related hardware message.

A hardware-message-details object with a **type** value of **"basic"** provides general details. In addition to the base properties, it contains the following type-specific properties:

*Table 431. hardware-message-details type-specific properties when the **type** value is **"basic"**:*		
Name	**Type**	**Description**
messages	Array of String	The detail messages for the hardware message.

A hardware-message-details object with a **type** value of **"common-problem"** provides problem details. In addition to the base properties, it contains the following type-specific properties:

*Table 432. hardware-message-details type-specific properties when the **type** value is **"common-problem"**:*		
Name	**Type**	**Description**
created	Timestamp	The timestamp representing when the problem was created.
description	Array of String	The problem description, or null if there is no description.
corrective-action	Array of String	The corrective action for the problem, or null if there is no corrective action.
repair-impact	Array of String	The repair impact for the problem, or null if there is no repair impact.
problem-data	Array of Object	An array of hardware-message-data-details objects for the problem data, or null if there is no problem data.
lir-node-data	Array of Object	An array of hardware-message-node-details objects for the problem Link Incident Record (LIR) node data, or null if there is no LIR node data. A Link Incident Record is reported by the Channel Subsystem for a hardware problem on an ESCON or FICON link.

A hardware-message-node-details object provides a title along with associated data detail:

Table 433. hardware-message-node-details:		
Name	**Type**	**Description**
title	String	The node title.
details	Array of Object	An array of hardware-message-data-details objects.

A hardware-message-data-details object provides simple data detail:

Table 434. hardware-message-data-details:		
Name	**Type**	**Description**
caption	String	The data caption.
value	String	The data value.

The mobile-app-preferences nested object contains the following fields:

Table 435. mobile-app-preferences object properties		
Name	**Type**	**Description**
app-enabled	Boolean	True indicates that the mobile app is enabled for this console. False indicates that the mobile app is disabled and will not be allowed to access this console.
require-app-password-enabled	Boolean	True indicates that the mobile app must require its user to set an app password in order to access the app itself. False indicates the user is not required to secure the app with a password.
password-caching-enabled	Boolean	True indicates that the mobile app is permitted to securely store the user's password on the device for this console. False indicates this is not permitted, and the user must input the password on every logon to this console from the app.
actions-enabled	Boolean	True indicates that the mobile app is permitted to perform actions against this console and the objects it manages, as permitted by the user's authority. False indicates that no actions are permitted from the mobile app to this console, and the app will be used only for monitoring.
action-settings-activate-partition	action-settings object	The mobile app settings information for the activate logical partition action.
action-settings-deactivate-partition	action-settings object	The mobile app settings information for the deactivate logical partition action.
action-settings-start-partition	action-settings object	The mobile app settings information for the start DPM partition action.
action-settings-stop-partition	action-settings object	The mobile app settings information for the stop DPM partition action.
action-settings-change-activation-profile	action-settings object	The mobile app settings information for the change activation profile of a logical partition action.
action-settings-load-os-into-partition	action-settings object	The mobile app settings information for the load OS into logical partition action.
action-settings-reset-partition	action-settings object	The mobile app settings information for the reset logical partition action.
action-settings-change-partition-weight	action-settings object	The mobile app settings information for the change DPM partition or logical partition weights and capping action.

Table 435. mobile-app-preferences object properties (continued)		
Name	**Type**	**Description**
action-settings-change-partition-processors	action-settings object	The mobile app settings information for the change DPM partition or logical partition processors action.
action-settings-delete-hardware-message	action-settings object	The mobile app settings information for the delete hardware message action.
action-settings-request-service-hardware-message	action-settings object	The mobile app settings information for the request service for a hardware message action.
action-settings-delete-os-message	action-settings object	The mobile app settings information for the delete OS message action.
action-settings-send-os-command	action-settings object	The mobile app settings information for the send OS command action.
action-settings-manage-remote-firmware-updates	action-settings object	The mobile app settings information for the manage remote firmware updates action.
notifications-enabled	Boolean	True indicates that notifications from this console to the mobile app are enabled and will flow to any registered devices. False indicates that notifications will not flow from this console to any device with the mobile app.
enhanced-notifications-enabled	Boolean	True indicates that enhanced notifications from this console to the mobile app are enabled and may flow to any registered devices. False indicates that enhanced notifications will not flow from this console to any device with the mobile app. [Added by feature **mobile-enhanced-push**]

The action-settings nested object contains the following fields:

Table 436. action-settings object properties		
Field name	**Type**	**Description**
enablement	String Enum	The value indicating the action enablement setting for the mobile app. Valid values: • **"disabled"** - The action is not allowed to be performed from the mobile app for this HMC, even if the global **actions-enabled** setting is **true**. • **"enabled"** - The action is allowed to be performed from the mobile app for this HMC, unless the global **actions-enabled** setting is **false**. • **"enabled-for-users"** - The action is allowed to be performed from the mobile app for this HMC if and only if the global **actions-enabled** setting is **true** and the user or pattern-based user's template is included in the **enabled-users-and-templates** property.

Table 436. action-settings object properties (continued)		
Field name	**Type**	**Description**
enabled-users-and-templates	Array of String/ URI	The canonical URI paths of Users and User Templates for which the action is enabled. This property is only included when the **enablement** property has a value of **"enabled-for-users"**. When included, its value must always have at least one URI. Note that this property cannot contain users of **type pattern-based**.

Table 437. hma-info object properties		
Name	**Type**	**Description**
peer-hmc	hma-peer-hmc-info object	Describes information about a peer HMA Hardware Management Console known to the local Hardware Management Console. If there is no known peer HMA Hardware Management Console, this property is null.
guests	Array of hma-guest-info objects	Describes information about the HMA guests hosted by the local Hardware Management Console. If there is no known guest, this property is null.

Every hma-guest-info object contains the following base properties:

Table 438. hma-guest-info object properties		
Name	**Type**	**Description**
type	String Enum	The type of guest being hosted. Valid values: • **"se"** - Support Element

A hma-guest-info object with a type value of "se" provides the following type specific properties in addition to the base properties:

*Table 439. hma-guest-info object type-specific properties when the type value is **"se"***		
Name	**Type**	**Description**
se-name	String (0-16)	The guest Support Element name or an empty string if the name is not known.
role	String Enum	An indication of the guest Support Element's (SE) role. One of the following values: Valid values: • **"primary"** - The primary SE • **"alternate"** - The alternate SE • **"unknown"** - Role not known

Remote Firmware Update Console element object

A Remote Firmware Update element object defines a firmware update operation that is scheduled to occur on the Console at a future time.

Table 440. Console object - Remote Firmware Update element object properties			
Name	**Qualifier**	**Type**	**Description**
element-id	—	String (36)	The unique identifier for the remote firmware update instance.
element-uri	—	String/ URI	The canonical URI path for the remote firmware update object, of the form `/api/console/remote-firmware-updates/{remote-firmware-update-id}`, where `{remote-firmware-update-id}` is the **element-id** of this remote firmware update.
parent	—	String/ URI	The parent of a remote firmware update is conceptually its owning console, and so the parent value is the canonical URI path for the console.
class	—	String (22)	The class of a Remote Firmware Update element is **"remote-firmware-update"**.
creation-time	—	Timestamp	The time at which the remote firmware update was scheduled.
scheduled-execution-time	—	Timestamp	The time at which the remote firmware update will begin.
execution-window	—	Integer	The number of minutes the operation will wait if it is blocked at its scheduled execution time, for example due to a busy condition.
execution-percentage	(pc)	Integer (0-100)	The current percentage of this firmware update operation that has completed. [Added by feature **rcl-progress**]
execution-steps	(pc)	Array of Remote Firmware Update Execution Step Console objects	Array of objects that describe the steps in this firmware update operation. [Added by feature **rcl-progress**]
target-bundle	—	String	The target bundle level for the firmware update.
backup-location	—	String Enum	The location of the backup data set. Values: • **"usb"** - The backup will be saved to a USB device that is mounted on the HMC. • **"ftp"** - The backup will be saved to an FTP server.

Table 440. Console object - Remote Firmware Update element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
state	(pc)	String Enum	The execution state of the scheduled operation. Values: • **"scheduled"** - The current time is not beyond the **"scheduled-execution-time"**. • **"running"** - The operation is currently executing. • **"scheduled-on-peer"** - The operation is currently in a **"scheduled"** state on the peer. This state is only applicable when the target of the operation is a console defined as an HMA. • **"running-on-peer"** - The operation is currently in a **"running"** state on the peer. This state is only applicable when the target of the operation is a console defined as an HMA. • **"pending"** - The operation is complete pending a condition. In order to compete the operation, an additional action specified within the **"pending-conditions"** property will have to be fulfilled. [Added by feature **rcl-history**] • **"succeeded"** - The operation completed successfully. If the target of the operation is a specified CPC, then the state value also indicates that the operation completed without any pending conditions. [Added by feature **rcl-history**] • **"failed"** - The operation failed. The **execution-steps** list should contain a Remote Firmware Update Execution Step Console object with a **state** value of **"failed"** to indicate the precise step of failure. [Added by feature **rcl-history**]
scheduling-console-name	—	String	Name of the console from which the remote firmware update was scheduled. [Added by feature **rcl-progress**]
service-contact-name	—	String	The name of the service representative that scheduled the operation. The value may be empty.
service-contact-telephone-number	—	String	The telephone number of the service representative that scheduled the operation. The value may be empty.
service-contact-email-address	—	String	The email address of the service representative that scheduled the operation. The value may be empty.

Remote Firmware Update Execution Step Console nested object

A Remote Firmware Update Execution Step Console nested object describes a step in the overall Console firmware update process. [Added by feature **rcl-progress**]

Table 441. Console object - Remote Firmware Update Execution Step Console nested object properties			
Name	**Qualifier**	**Type**	**Description**
id	—	String Enum	Identifies the step whose progress is described by this object Values • **"verify-environment"** - Checks that change management is enabled and other conditions that would cause the remote code load to fail. • **"back-up-critical-data"** - Makes a backup of the targeted platform so that if the 1U server needs to be replaced, it can be restored from this data. • **"accept-installed-changes"** - Makes the previously activated bundle permanent so that it cannot be backed off. • **"retrieve-internal-code-changes"** - Pulls the latest internal code changes that have been released to zRSF. • **"apply-internal-code-changes"** - Activates the internal code changes up to the bundle requested in the remote code load. • **"transmit-system-availability-data"** - Collects data including system status and send it back to the remote support system.
state	—	String Enum	The execution state of the firmware update step. The progression of states for a particular execution step should be **"not-started"**, then **"running"**, and then finally to either **"succeeded"** or **"failed"**. The execution state for a particular execution step will remain as **"not-started"** if a failure occurred at a previous execution step. Values: • **"not-started"** - The step has not started yet. • **"running"** - The step is currently executing. • **"succeeded"** - The step completed successfully. • **"failed"** - The step failed.

Get Console Properties

The `Get Console Properties` operation retrieves the properties of the Console object. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console
```

Query Parameters

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the Console object's Data Model.

Response body contents

On successful completion, the response body contains an object that provides the current values of the properties for the Console object as defined in "Data model" on page 801. Field names and data types in the object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the Console object.

On successful execution, HTTP status code 200 (OK) is returned and all of the current properties as defined by the data model for the Console object are provided in the response body.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, all authenticated users have permission.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described "Response body contents" on page 812.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console HTTP/1.1
x-api-session: duqt0x27jl7a9sn50e9w2sh5f0nxitb56zbiqnc5yjwxrsvvz
```

Figure 402. Get Console Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 22:04:21 GMT
content-type: application/json;charset=UTF-8
content-length: 4397
{
   "class":"console",
   "description":"",
   "ec-mcl-description":{
      "ec":[
         {
            "description":"Hardware Management Console Framework",
            "mcl":[
               {
                  "last-update":null,
                  "level":"-",
                  "type":"retrieved"
               },
               {
                  "last-update":null,
                  "level":"-",
                  "type":"activated"
               },
               {
                  "last-update":null,
                  "level":"-",
                  "type":"accepted"
               },
               {
                  "last-update":null,
                  "level":"-",
                  "type":"installable-concurrent"
               },
               {
                  "last-update":null,
                  "level":"-",
                  "type":"removable-concurrent"
               }
            ],
            "number":"P46683",
            "part-number":"02WG827",
            "type":"SYSTEM"
         },
```

Figure 403. Get Console Properties: Response (Part 1)

```
        {
           "description":"Hardware Management Console Platform Firmware",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46658",
           "part-number":"02WF276",
           "type":"HMCBIOS"
        },
        {
           "description":"Licensed Internal Code Alerts",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46684",
           "part-number":"02WG828",
           "type":"MALERT"
        },
```


Figure 404. Get Console Properties: Response (Part 2)

```
        {
           "description":"Enablement of new features   ",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46685",
           "part-number":"02WG829",
           "type":"ENABLE1"
        },
        {
           "description":"Enablement of new features   ",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46686",
           "part-number":"02WG830",
           "type":"ENABLE2"
        },
```


Figure 405. Get Console Properties: Response (Part 3)

```
        {
           "description":"Firmware feature enablement  ",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46687",
           "part-number":"02WG831",
           "type":"FFE"
        },
        {
           "description":"Open source components",
           "mcl":[
              {
                 "last-update":null,
                 "level":"-",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"-",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46688",
           "part-number":"02WG832",
           "type":"OPENSRC"
        },
```


Figure 406. Get Console Properties: Response (Part 4)

```
            {
               "description":"Embedded Operating System",
               "mcl":[
                  {
                     "last-update":null,
                     "level":"-",
                     "type":"retrieved"
                  },
                  {
                     "last-update":null,
                     "level":"-",
                     "type":"activated"
                  },
                  {
                     "last-update":null,
                     "level":"-",
                     "type":"accepted"
                  },
                  {
                     "last-update":null,
                     "level":"-",
                     "type":"installable-concurrent"
                  },
                  {
                     "last-update":null,
                     "level":"-",
                     "type":"removable-concurrent"
                  }
               ],
               "number":"P45842",
               "part-number":"02WG826",
               "type":"OS"
            }
         ]
      },
      "is-locked":false,
      "machine-info":{
         "machine-model":"TW2",
         "machine-serial":"DK5C004",
         "machine-type":"2461"
      },
      "name":"HMCDAILY03",
      "network-info":{
         "this-hmc":[
            {
               "domain-name":"local",
               "hmc-name":"HMCDAILY03",
               "interface-name":"eth0",
               "ipv4-address":[
                  {
                     "ip-address":"192.0.2.0",
                     "subnet-mask":"255.255.255.0"
                  }
               ],
               "ipv6-address":[
                  {
                     "ip-address":"2001:0db8:0:0:0:210:6fff:9759",
                     "prefix-length":64
                  }
               ],
               "is-private":false,
               "mac":"00106f0d9759"
            },
```


Figure 407. Get Console Properties: Response (Part 5)

```
            {
               "domain-name":"local",
               "hmc-name":"HMCDAILY03",
               "interface-name":"eth1",
               "ipv4-address":[
                  {
                     "ip-address":"0.0.0.0",
                     "subnet-mask":"255.255.255.255"
                  }
               ],
               "ipv6-address":[],
               "is-private":false,
               "mac":"00106f0d975a"
            },
            {
               "domain-name":"local",
               "hmc-name":"HMCDAILY03",
               "interface-name":"eth2",
               "ipv4-address":[
                  {
                     "ip-address":"0.0.0.0",
                     "subnet-mask":"255.255.255.0"
                  }
               ],
               "ipv6-address":[],
               "is-private":false,
               "mac":"00106f0d975b"
            },
            {
               "domain-name":"local",
               "hmc-name":"HMCDAILY03",
               "interface-name":"eth3",
               "ipv4-address":[
                  {
                     "ip-address":"0.0.0.0",
                     "subnet-mask":"255.255.255.0"
                  }
               ],
               "ipv6-address":[],
               "is-private":false,
               "mac":"00106f0d975c"
            }
         ]
      },
      "object-id":"106ffdd7-04fa-376c-a059-6c69486bc57f",
      "object-uri":"/api/console",
      "parent":null,
      "version":"2.15.0"
   }
```


Figure 408. Get Console Properties: Response (Part 6)

Restart Console

The `Restart Console` operation restarts the Hardware Management Console. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/console/operations/restart
```

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/Opt	Description
`force`	Boolean	Optional	Whether the restart operation is processed when users are connected (true) or not (false). The default is false.

Description

The Console is restarted.

By default, the restart does not occur if one or more users are currently connected to the Console. This can be overridden by use of the **force** field in the request body.

On success, HTTP status code 202 (Accepted) is returned.

Authorization

To use `Restart Console`, you must have the following:

- For the web services interface, action/task permission to the **Power Off or Restart** task
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.
- Remote Restart must be enabled on the Hardware Management Console or Support Element.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See"Common request validation reason codes" on page 66 for a list of the possible reason codes.
	267	The operation is rejected, due to the presence of HMC users. Either wait until all HMC users have logged off or retry the request with the **force** field set to **"true"**.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	269	This operation is currently blocked. The error message will contain information on the blocking application.
	270	The remote restart operation is not enabled on the HMC.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Shutdown Console

`Shutdown Console` powers off the Hardware Management Console. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/console/operations/shutdown
```

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/Opt	Description
`force`	Boolean	Optional	Whether the shutdown operation is processed when users are connected (true) or not (false). The default is false.

Description

The Console is powered off.

By default, the shutdown does not occur if one or more users are currently connected to the Console. This can be overridden by use of the force field in the request body.

The action to shutdown the Console occurs asynchronously. If the request is accepted, HTTP status code 202 (Accepted) is returned to indicate that the request has been initiated. However, because this action results in the targeted Console becoming inactive and powered off at completion, it is not possible to track the completion of this request. Thus no response body containing an asynchronous job URI is provided, nor is a job completion notification generated upon completion.

Authorization

To use `Shutdown Console`, you must have the following:

- For the web services interface, action/task permission to the **Power Off or Restart** task
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.
- Remote Shutdown must be enabled on the Hardware Management Console or Support Element.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned but no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	267	The operation is rejected, due to the presence of HMC users. Either wait until all HMC users have logged off or retry the request with the force field set to true.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	270	The remote restart operation is not enabled on the HMC.
	304	This operation is currently blocked. The error message will contain information on the blocking application.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/operations/shutdown HTTP/1.1/
x-api-session: 5dul8zvlwa5s83eobcukaf1vug3s3kgidkyk9e5c5acsekabsl
content-type: application/json
content-length: 16
{
   "force": false
}
```

Figure 409. Shutdown Console: Request

```
202 Accepted
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 01 Mar 2013 19:38:25 GMT

<No response body>
```

Figure 410. Shutdown Console: Response

Reorder User Patterns

The `Reorder User Patterns` operation changes the search order of the console's User Patterns used when a user logs on to the console.

HTTP method and URI

```
POST /api/console/operations/reorder-user-patterns
```

Request body contents

The request body is a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`user-pattern-uris`	Array of String/URI	Required	Ordered list of User Pattern object **element-uri** property values. The order of these URIs in the array defines the new order of the User Patterns.

Description

This operation reorders the console's User Patterns.

On successful execution of this operation the User Patterns are reordered to match the order of their **element-uri** properties in the **user-pattern-uris** array in the request body.

The request body is validated against the schema described in "Request body contents" on page 822. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If a URI in the request body does not designate an existing User Pattern object, status code 404 (Not Found) is returned. The array in the request body must include each of the console's currently defined User Patterns and no others. In addition, the API user must have action/task permission to the Manage User Patterns task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirement

This operation has the following authorization requirements:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 442. Reorder User Patterns: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The array in the request body is missing an existing User Pattern, or it contains an entry that designates a User Pattern that is not one of the console's current User Patterns.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	2	A URI in the request body does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/reorder-user-patterns HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 301
{
   "user-pattern-uris":[
      "/api/console/user-patterns/497bf4ec-1dbf-11e4-8ceb-1c6f65065a91",
      "/api/console/user-patterns/6d897292-3ceb-11e4-9e36-1c6f65065a91",
      "/api/console/user-patterns/e40b9ba6-48e0-11e4-82a1-1c6f65065a91",
      "/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91"
   ]
}
```

Figure 411. Reorder User Patterns: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 412. Reorder User Patterns: Response

Get Console Audit Log

The `Get Console Audit Log` operation returns the console audit log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/operations/get-audit-log
```

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

A log-entry-info object contains information about a single log entry and the event which caused the entry. Each log-entry-info object contains the following fields:

Table 443. log-entry-info object properties

Name	Type	Description
event-time	Timestamp	The time when the event occurred.
event-id	String	The ID number for the event.
event-name	String (0-12)	The name for the event, or null or an empty string if none.
userid	String	The user ID of the HMC/SE user associated with the event, or null if there is no user associated with the event.
user-uri	String/ URI	The canonical URI path of the HMC/SE user associated with the event, or null is there is no user associated with the event.
event-message	String	The complete, formatted message for the event.
event-data-items	Array of objects	An array of event-data-item-info objects, one for each item of event data associated with the **event-message** for the event. If there are no event data items, an empty array is provided. The order of items in this array is semantically significant and matches the documentation for this **event-id**.
event-details	Array of objects	An array of event-details-info objects, one for each event detail associated with the event. If there are no event details, an empty array is provided.

An event-details-info object contains information about a single event detail. Each event-details-info object contains the following fields:

Table 444. event-details-info object properties

Name	Type	Description
event-details-message	String	The complete, formatted details message.

Table 444. event-details-info object properties (continued)		
Name	**Type**	**Description**
event-details-data-items	Array of objects	An array of event-data-item-info objects, one for each item of event details data associated with the **event-details-message** for the event. If there are no event details data items, an empty array is provided.

An event-data-item-info object contains information about a single item of event data. Each event-data-item-info object contains the following fields:

Table 445. event-data-item-info object properties		
Name	**Type**	**Description**
data-item-number	Integer	The number for this data item. This is the 0-based index of this event-data-item-info object in the event-data-items or event-details-data-items array in which it is contained. This number identifies the substitution variable to which this data item corresponds in the documentation for the event identified by **event-id**.
data-item-type	String Enum	Identifies the data type of the data item in the event-data-item field. Possible values are: • **"long"** • **"float"** • **"string"**
data-item-value	Varies. See **data-item-type** description	The data item.

Description

This operation returns the console's audit log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the begin-time query parameter is specified, then any entries earlier than that time are omitted. If the end-time query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

For the web services interface the API user must have action/task permission to the **Audit and Log Management** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the audit log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirements:

- For the web services interface, action/task permission to the **Audit and Log Management** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described "Response body contents" on page 825.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-audit-log HTTP/1.1
x-api-session: 3ws0pztvnx61quwkvk4kcpjodybuqyo8q840j2adzw8y08fglh
```

Figure 413. Get Console Audit Log: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Wed, 18 Sep 2019 22:26:52 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"192.0.2.0"
         },
         {
            "data-item-number":1,
            "data-item-type":"string",
            "data-item-value":"403"
         },
         {
            "data-item-number":2,
            "data-item-type":"string",
            "data-item-value":"Logon"
         },
         {
            "data-item-number":3,
            "data-item-type":"string",
            "data-item-value":"SYSTEM"
         },
         {
            "data-item-number":4,
            "data-item-type":"string",
            "data-item-value":"Login failure, bad credentials for userid 8675309"
         },
         {
            "data-item-number":5,
            "data-item-type":"string",
            "data-item-value":"/api/sessions"
         }
      ],
      "event-details":[],
      "event-id":"6055",
      "event-message":"A web services client on 192.0.2.0 attempted an unauthorized (403)
         action \"Logon\" as SYSTEM: Login failure, bad credentials for userid 8675309
         (URI:/api/sessions)",
      "event-name":"WSAPI",
      "event-time":1487964584530,
      "user-uri":null,
      "userid":null
   },
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"192.0.2.0"
         },
         {
            "data-item-number":1,
            "data-item-type":"string",
            "data-item-value":"403"
         },
         {
            "data-item-number":2,
            "data-item-type":"string",
            "data-item-value":"Set Mobile App Preferences"
         },
         {
            "data-item-number":3,
            "data-item-type":"string",
            "data-item-value":"pedebug"
         },
```

Figure 414. Get Console Audit Log: Response (Part 1)

```
         {
            "data-item-number":4,
            "data-item-type":"string",
            "data-item-value":"Console"
         },
         {
            "data-item-number":5,
            "data-item-type":"string",
            "data-item-value":"HMCDAILY03"
         },
         {
            "data-item-number":6,
            "data-item-type":"string",
            "data-item-value":"Task"
         },
         {
            "data-item-number":7,
            "data-item-type":"string",
            "data-item-value":"Mobile App Preferences"
         },
         {
            "data-item-number":8,
            "data-item-type":"string",
            "data-item-value":"/api/console/operations/set-mobile-app-preferences"
         }
      ],
      "event-details":[],
      "event-id":"6053",
      "event-message":"A web services client on 192.0.2.0 attempted an unauthorized (403)
action \"Set Mobile App Preferences\" as pedebug against the Console object named
\"HMCDAILY03\".  User does not have permission to the Task named \"Mobile App Preferences\"
(URI:/api/console/operations/set-mobile-app-preferences)",
      "event-name":"WSAPI",
      "event-time":1488213109380,
      "user-uri":"/api/users/0540e45c-686a-11e6-852d-00106f0d5d80",
      "userid":"PEDEBUG"
   },
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"192.0.2.0"
         },
         {
            "data-item-number":1,
            "data-item-type":"string",
            "data-item-value":"403"
         },
         {
            "data-item-number":2,
            "data-item-type":"string",
            "data-item-value":"GET"
         },
         {
            "data-item-number":3,
            "data-item-type":"string",
            "data-item-value":"n/a"
         },
         {
            "data-item-number":4,
            "data-item-type":"string",
            "data-item-value":"Asserted cradentials did not map to a session"
         },
         {
            "data-item-number":5,
            "data-item-type":"string",
            "data-item-value":"/api/console/operations/get-mobile-app-preferences"
         }
      ],
```


Figure 415. Get Console Audit Log: Response (Part 2)

```
      "event-details":[],
      "event-id":"6055",
      "event-message":"A web services client on 192.0.2.0 attempted an unauthorized (403)
action \"GET\" as n/a: Asserted cradentials did not map to a session (URI:/api/console/
operations/get-mobile-app-preferences)",
      "event-name":"WSAPI",
      "event-time":1567620717940,
      "user-uri":null,
      "userid":null
   }
]
```

Figure 416. Get Console Audit Log: Response (Part 3)

Get Console Security Log

The `Get Console Security Log` operation returns the console security log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/operations/get-security-log
```

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

Description

This operation returns the console's security log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the begin-time query parameter is specified, then any entries earlier than that time are omitted. If the end-time query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

For the web services interface the API user must have action/task permission to the **View Security Logs** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirements:

- For the web services interface, action/task permission to the **View Security Logs** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described “Response body contents” on page 831.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-security-log HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 417. Get Console Security Log: Request


```
200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"acsadmin"
         },
         {
            "data-item-number":1,
            "data-item-type":"string",
            "data-item-value":"Sx149"
         },
         {
            "data-item-number":2,
            "data-item-type":"string",
            "data-item-value":"admin.my.company.com [1.2.3.4]"
         }
      ],
      "event-details":[],
      "event-id":"1941",
      "event-message":"User acsadmin has logged on to Web Services API session
         Sx149 from location admin.my.company.com [1.2.3.4]",
      "event-name":"WSA Logon",
      "event-time":1412285249660,
      "user-uri":"/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
      "userid":"ACSADMIN"
   },
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"Dept Admin"
         }
      ],
      "event-details":[
         {
            "event-details-data-items":[],
            "event-details-message":"Task Role: Dept Admin  --Based on role is
               null. Permitted tasks:"
         }
      ],
      "event-id":"1272",
      "event-message":"The task role Dept Admin has been created.",
      "event-name":"LOGTROLEADD",
      "event-time":1412285252280,
      "user-uri":"/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
      "userid":"ACSADMIN"
   },
```


Figure 418. Get Console Security Log: Response (Part 1)

```
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"Dept Admin"
         }
      ],
      "event-details":[
         {
            "event-details-data-items":[],
            "event-details-message":"Task Role: Dept Admin  --Based on role is
               null. Permitted tasks:  ClassId=XVirtualServer"
         }
      ],
      "event-id":"1273",
      "event-message":"The task role Dept Admin has been changed.",
      "event-name":"LOGTROLECHG",
      "event-time":1412285253000,
      "user-uri":"/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
      "userid":"ACSADMIN"
   }
]
```


Figure 419. Get Console Security Log: Response (Part 2)

Get Console Events Log

The `Get Console Events Log` operation returns the console events log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/operations/get-events-log
```

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

Description

This operation returns the console's events log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

For the web services interface the API user must have action/task permission to the **View Console Events** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirements:

- For the web services interface, action/task permission to the **View Console Events** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described “Response body contents” on page 834.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-events-log HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 420. Get Console Events Log: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Sat, 06 Mar 2021 19:19:44 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items": [
         {
            "data-item-number": 0,
            "data-item-type": "string",
            "data-item-value": "acsadmin"
         },
         {
            "data-item-number": 1,
            "data-item-type": "long",
            "data-item-value": 0
         },
         {
            "data-item-number": 2,
            "data-item-type": "long",
            "data-item-value": 0
         },
         {
            "data-item-number": 3,
            "data-item-type": "string",
            "data-item-value": ""
         },
         {
            "data-item-number": 4,
            "data-item-type": "string",
            "data-item-value": "2"
         },
         {
            "data-item-number": 5,
            "data-item-type": "string",
            "data-item-value": "Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/
              20100101 Firefox/60.0"
         }
      ],
      "event-details": [],
      "event-id": "1408",
      "event-message": "User acsadmin has logged on from the console to session id 2.",
      "event-name": "Logon",
      "event-time": 1615058307630,
      "user-uri": "/api/users/e84a7562-7eac-11eb-91a2-fa163e79b388",
      "userid": "ACSADMIN"
   },
   {
      "event-data-items": [
         {
            "data-item-number": 0,
            "data-item-type": "string",
            "data-item-value": "\"Tip of the Day\""
         },
         {
            "data-item-number": 1,
            "data-item-type": "string",
            "data-item-value": "1"
         },
         {
            "data-item-number": 2,
            "data-item-type": "string",
            "data-item-value": "acsadmin"
         }, ,
```

Figure 421. Get Console Events Log: Response (Part 1)

```
        {
           "data-item-number": 3,
           "data-item-type": "string",
           "data-item-value": "2"
        },
        {
           "data-item-number": 4,
           "data-item-type": "string",
           "data-item-value": ""
        }
     ] "event-details": [],
     "event-id": "1989",
     "event-message": "Task \"Tip of the Day\" with identifier 1 started by
       user acsadmin in session 2.",
     "event-name": "TaskStart",
     "event-time": 1615058308020,
     "user-uri": "/api/users/e84a7562-7eac-11eb-91a2-fa163e79b388",
     "userid": "ACSADMIN"
  },
  {
     "event-data-items": [
        {
           "data-item-number": 0,
           "data-item-type": "string",
           "data-item-value": "\"Tip of the Day\""
        },
        {
           "data-item-number": 1,
           "data-item-type": "string",
           "data-item-value": "1"
        },
        {
           "data-item-number": 2,
           "data-item-type": "string",
           "data-item-value": "acsadmin"
        }
     ],
     "event-details": [],
     "event-id": "1991",
     "event-message": "Task \"Tip of the Day\" with identifier 1 for
       user acsadmin has ended.",
     "event-name": "TaskEnd",
     "event-time": 1615058320070,
     "user-uri": "/api/users/e84a7562-7eac-11eb-91a2-fa163e79b388",
     "userid": "ACSADMIN"
  },
```


Figure 422. Get Console Events Log: Response (Part 2)

```
    {
       "event-data-items": [
          {
             "data-item-number": 0,
             "data-item-type": "string",
             "data-item-value": "\"Customize API Settings\""
          },
          {
             "data-item-number": 1,
             "data-item-type": "string",
             "data-item-value": "2"
          },
          {
             "data-item-number": 2,
             "data-item-type": "string",
             "data-item-value": "acsadmin"
          },
          {
             "data-item-number": 3,
             "data-item-type": "string",
             "data-item-value": "2"
          },
          {
             "data-item-number": 4,
             "data-item-type": "string",
             "data-item-value": ""
          }
       ],
       "event-details": [],
       "event-id": "1989",
       "event-message": "Task \"Customize API Settings\" with identifier 2
         started by user acsadmin in session 2.",
       "event-name": "TaskStart",
       "event-time": 1615058327740,
       "user-uri": "/api/users/e84a7562-7eac-11eb-91a2-fa163e79b388",
       "userid": "ACSADMIN"
    },
    {
       "event-data-items": [
          {
             "data-item-number": 0,
             "data-item-type": "string",
             "data-item-value": "\"Customize API Settings\""
          },
          {
             "data-item-number": 1,
             "data-item-type": "string",
             "data-item-value": "2"
          },
          {
             "data-item-number": 2,
             "data-item-type": "string",
             "data-item-value": "acsadmin"
          }
       ],
       "event-details": [],
       "event-id": "1991",
       "event-message": "Task \"Customize API Settings\" with identifier 2
         for user acsadmin has ended.",
       "event-name": "TaskEnd",
       "event-time": 1615058352740,
       "user-uri": "/api/users/e84a7562-7eac-11eb-91a2-fa163e79b388",
       "userid": "ACSADMIN"
    }
]
```


Figure 423. Get Console Events Log: Response (Part 3)

List Console Hardware Messages

The `List Console Hardware Messages` operation lists the current set of hardware messages associated with the console. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/hardware-messages
```

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created earlier than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created later than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`hardware-messages`	Array of hardware-message-info objects	Array of nested hardware-message-info objects as defined in the next table.

Each nested hardware-message-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The canonical URI path of the hardware message. The URI is in the following form: `/api/console/hardware-messages/{hardware-message-id}`
`timestamp`	Timestamp	The date and time the hardware message was created
`text`	String	The text of the hardware message.

Description

This operation returns a set of console hardware messages in increasing timestamp order, filtered according to the query parameters, if specified.

If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

If there are no hardware messages associated with the console, or if no hardware messages are to be included in the results due to filtering, an empty array is returned and the operation completes successfully.

For the web services interface the API user must have Action/Task permission to the Hardware Messages task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, action/task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 839.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 446. List Console Hardware Messages: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The **begin-time** value is greater than the **end-time** value.
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/hardware-messages HTTP/1.1
x-api-session: 4pw149l9jtzcwohdfe8s9gw5zzv7v73yksomswrg50t7ni4q8r
```

Figure 424. List Console Hardware Messages: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Oct 2014 17:02:37 GMT
content-type: application/json;charset=UTF-8
content-length: 206
{
   "hardware-messages":[
      {
         "element-uri":"/api/console/hardware-messages/11c5f16a-4d58-11e4-ba8d-
            02215e673710",
         "text":"Licensed internal code has detected a problem.   [Problem # 3]",
         "timestamp":1412600137860
      }
   ]
}
```

Figure 425. List Console Hardware Messages: Response

Get Console Hardware Message Properties

The `Get Console Hardware Message Properties` operation retrieves the properties of a single console hardware message. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/hardware-messages/{hardware-message-id}
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message to be retrieved.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the console hardware message object as defined in "Data model" on page 801. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation retrieves the properties of a single console hardware message specified by *{hardware-message-id}*.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface action/task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 841.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
404 (Not Found)	322	The element ID in the URI *{hardware-message-id}* does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7 HTTP/1.1
x-api-session: 38v3uicf04oxah52q8spe8evm4esyr83andorav2rv2xe6flq9
```

Figure 426. Get Console Hardware Message Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:38:29 GMT
content-type: application/json;charset=UTF-8
content-length: 821
{
   "class":"hardware-message",
   "details":{
      "corrective-action":[
         "Service is required."
      ],
      "created":1583173801902,
      "description":[
         "The Hardware Management Console licensed internal code has detected a problem."
      ],
      "lir-node-data":null,
      "problem-data":[
         {
            "caption":"System name",
            "value":"Local"
         },
         {
            "caption":"Date",
            "value":"Mar 2, 2020"
         },
         {
            "caption":"Time",
            "value":"1:30:01 PM"
         }
      ],
      "repair-impact":[
         "The S/390 microprocessor cluster will continue operating, but some Hardware
            Management Console functions may not be available."
      ],
      "type":"common-problem"
   },
   "element-id":"ef3d8ce0-5cb3-11ea-b607-fa163e63cff7",
   "element-uri":"/api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7",
   "parent":"/api/console",
   "service-supported":true,
   "text":"Licensed internal code has detected a problem.   [Problem # 3]",
   "timestamp":1583173847091
```

Figure 427. Get Console Hardware Message Properties: Response

Delete Console Hardware Message

The `Delete Console Hardware Message` operation deletes a single console hardware message. This operation is supported using the BCPii interface.

HTTP method and URI

```
DELETE /api/console/hardware-messages/{hardware-message-id}
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message to be deleted.

Description

This operation deletes a specific console hardware message. The hardware message to be deleted is identified by the *{hardware-message-id}* variable in the URI.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface action/task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	322	The element ID in the URI *{hardware-message-id}* does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/console/hardware-messages/6b2d61a4-a1ac-11e4-87ee-5ef3fcae8020 HTTP/1.1
x-api-session: c8un3odpy8yyp150o3poz1ud4gwyfodlwyq495327bpyn2p0z
```

Figure 428. Delete Console Hardware Message: Request

```
204 No Content
date: Mon, 09 Feb 2015 20:07:31 GMT
server: zSeries management console API web server / 2.0

<No response body>
```

Figure 429. Delete Console Hardware Message: Response

Request Console Service

The `Request Console Service` operation electronically transmits problem information to request service for the error and deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/console/hardware-messages/{hardware-message-id}/operations/request-service
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message for which to request service.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`customer-name`	String	Optional	The name of the person that can be contacted about the problem.
`customer-phone`	String	Optional	The telephone number of the person that can be contacted about the problem.

Description

This operation electronically requests service for the problem reported by this hardware message. Customer contact information may optionally be provided in the request body. Upon successful completion, the service request is prepared and queued for transmission, and this hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. Remotely requesting service must also be enabled and configured on the HMC; otherwise, status code 409 (Conflict) is returned. In addition, for the web services interface the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface action/task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a console hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	322	The element ID in the URI *{hardware-message-id}* does not designate an existing console hardware message.
409 (Conflict)	341	The HMC is not enabled and configured for remotely requesting service.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/hardware-messages/ef3d8ce0-5cb3-11ea-b607-fa163e63cff7/operations/
   request-service HTTP/1.1
x-api-session: fvnsnbxrbi13lmzk7poonx8upx515zbq4qyz0ghf8z7yxiev4
content-type: application/json
content-length: 56
{
   "customer-name":"Jenny",
   "customer-phone":"867-5309"
}
```

Figure 430. Request Console Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:46:47 GMT

<No response body>
```

Figure 431. Request Console Service: Response

Get Console Service Request Information

The `Get Console Service Request Information` operation returns problem information and a telephone number to be used for requesting service for the error and optionally deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/hardware-messages/{hardware-message-id}/operations/get-service-information
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message for which to request service.

Query parameters

Name	Type	Rqd/Opt	Description
`delete`	Boolean	Optional	A value of **true** will delete the hardware message upon successful completion, **false** will not delete the hardware message. Default: **true**

Response body contents

On successful completion, the response body contains a JSON object with the following fields.

Name	Type	Description
service-phone	String	The telephone number to call for service.
machine-model	String (1-3)	The model of the machine where the problem occurred.
machine-type	String (1-4)	The type of the machine where the problem occurred.

Name	Type	Description
machine-serial-number	String (1-12)	The serial number of the machine where the problem occurred.
problem-type	String	The type of the problem to be reported to service.
problem-number	Integer	The identifying number of the problem to be reported to service.
problem-data	String	Additional problem data to be reported to service.
reference-code	String	The problem reference code to be reported to service.
customer-name	String	The name of the administrator of the HMC. This field will be omitted if the API user does not have authority to the **Customize Customer Information** task.
customer-phone	String	The telephone number of the administrator of the HMC. This field will be omitted if the API user does not have authority to the **Customize Customer Information** task.

Description

This operation is used to manually request service for the problem reported by this hardware message. This may be used if remote service is not configured or not functioning to call a service representative directly and provide the problem details. Upon successful completion, problem details to be reported to service are returned, and if the **delete** query parameter is **true**, this hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, for the web services interface the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface action/task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 846.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a console hardware message that does not support service.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	322	The element ID in the URI *{hardware-message-id}* does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/console/hardware-messages/7ed7e20e-5d7e-11ea-b607-fa163e63cff7/operations/
   get-service-information HTTP/1.1
x-api-session: 35wpq9m1ab0yxi9pvx27n673khzeajtt9c2v6fz2vsrf9g3ex0
```

Figure 432. Get Console Service Request Information: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:50:25 GMT
content-type: application/json;charset=UTF-8
content-length: 210
{
   "machine-model":"T01",
   "machine-serial-number":"000020026EA8",
   "machine-type":"8561",
   "problem-data":"",
   "problem-number":4,
   "problem-type":"2",
   "reference-code":"E5D43206-BD254E4E",
   "service-phone":"1-800-IBM-SERV"
}
```

Figure 433. Get Console Service Request Information: Response

Decline Console Service

The `Decline Console Service` operation declines service for the error and deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/console/hardware-messages/{hardware-message-id}/operations/decline-service
```

In this request, the URI variable *{hardware-message-id}* is the unique identifier of the hardware message for which to decline service.

Description

This operation is used to decline service for the problem reported by this hardware message. Upon successful completion, the hardware message is deleted.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, for the web services interface the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned. For

the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface action/task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a console hardware message that does not support service.
403 (Forbidden)	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	322	The element ID in the URI *{hardware-message-id}* does not designate an existing console hardware message.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/hardware-messages/54efcda6-5cb3-11ea-b607-fa163e63cff7/operations/
   decline-service HTTP/1.1
x-api-session: 4ya8pci1s5u4jt0g9dvjilt7bpwcpb29nhkgnt2c63uan87upp
content-type: application/json
```

Figure 434. Decline Console Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:53:05 GMT

<No response body>
```

Figure 435. Decline Console Service: Response

List Unmanaged CPCs

The `List Unmanaged CPCs` operation lists the CPCs that have been discovered by this HMC but are not configured to be managed by this HMC.

HTTP method and URI

```
GET /api/console/operations/list-unmanaged-cpcs
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching name.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
cpcs	Array of cpc-info objects	Array of nested cpc-info objects as described in the next table.

Each nested cpc-info object contains the following fields:

Field name	Type	Description
object-uri	String/ URI	The canonical URI path for an unmanaged CPC object is of the form `/api/cpcs/{cpc-id}` where *{cpc-id}* is the unique identifier for the unmanaged CPC instance.
name	String	The name of the unmanaged CPC.

Description

This operation lists the CPCs that have been discovered by this HMC but are not configured to be managed by this HMC. Some basic information is provided for each CPC that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those unmanaged CPCs whose names match the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

An unmanaged CPC is included in the list only if the API user has object-access permission to that object. If there is an unmanaged CPC to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no unmanaged CPCs known to the HMC or if no unmanaged CPCs are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the unmanaged CPC objects included in the response body.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 850.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 447. List Unmanaged CPCs: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage Notes

The APIs provide only minimal support for unmanaged CPC objects. There is no data model, and most of the standard operations are not provided. The `List Unmanaged CPCs` operation can be used to list them and to provide their name and object URI. The provided support also allows an API client to manage access to them through the `Add Permission to User Role` and `Remove Permission from User Role` operations. The HMC's **Add Object Definition** task is used to configure an unmanaged CPC to be managed by the HMC.

Example HTTP interaction

```
GET /api/console/operations/list-unmanaged-cpcs HTTP/1.1
x-api-session: 606ay5h8erhjmee80hl7j4rglorqebifiirqnrd4m1ga9xyjv6
```

Figure 436. List Unmanaged CPCs: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 14:13:05 GMT
content-type: application/json;charset=UTF-8
content-length: 252
{
   "cpcs":[
      {
         "name":"P0LXSM20",
         "object-uri":"/api/cpcs/343a56e1-96ed-3191-b092-99a784311e43"
      },
      {
         "name":"D20BUSE",
         "object-uri":"/api/cpcs/07cc8420-78e6-3520-840a-ea6ff0074bbc"
      },
      {
         "name":"S202B",
         "object-uri":"/api/cpcs/3275e681-fe9b-3c54-ade8-6a08b802f781"
      }
   ]
}
```

Figure 437. List Unmanaged CPCs: Response

Get Mobile App Preferences

The `Get Mobile App Preferences` operation provides the current preferences for the mobile app on this console.

HTTP method and URI

```
GET /api/console/operations/get-mobile-app-preferences
```

Response body contents

On successful completion, the response body contains a JSON object that provides the **mobile-app-preferences** property of the Console object as defined in "Data model" on page 801.

Description

This operation returns the **mobile-app-preferences** property of the Console object.

On successful execution, the **mobile-app-preferences** property is provided in the response body, and HTTP status code 200 (OK) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 852.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/get-mobile-app-preferences HTTP/1.1
x-api-session: 51hgmqbgce40n9lgt40g3tz5ktlsjqtn4hz8ng2z7m0npyb9lu
```

Figure 438. Get Mobile App Preferences: Request

```
200
Content-Type: application/json;charset=UTF-8
Content-Length: 1045
{
   "mobile-app-preferences":{
      "action-settings-activate-partition":{
         "enablement":"enabled"
      },
      "action-settings-change-activation-profile":{
         "enablement":"enabled"
      },
      "action-settings-change-partition-processors":{
         "enablement":"enabled"
      },
      "action-settings-change-partition-weight":{
         "enablement":"enabled"
      },
      "action-settings-deactivate-partition":{
         "enablement":"enabled"
      },
      "action-settings-delete-hardware-message":{
         "enablement":"enabled"
      },
      "action-settings-delete-os-message":{
         "enablement":"enabled"
      },
      "action-settings-load-os-into-partition":{
         "enablement":"enabled"
      },
      "action-settings-request-service-hardware-message":{
         "enablement":"enabled"
      },
      "action-settings-reset-partition":{
         "enablement":"enabled"
      },
      "action-settings-send-os-command":{
         "enablement":"enabled"
      },
      "action-settings-start-partition":{
         "enablement":"enabled"
      },
      "action-settings-stop-partition":{
         "enablement":"enabled"
      },
      "actions-enabled":true,
      "app-enabled":true,
      "enhanced-notifications-enabled":false,
      "notifications-enabled":true,
      "password-caching-enabled":true,
      "require-app-password-enabled":false
   }
}
```

Figure 439. Get Mobile App Preferences: Response

Set Mobile App Preferences

The `Set Mobile App Preferences` operation sets one or more of the mobile app preferences for this console.

HTTP method and URI

```
POST /api/console/operations/set-mobile-app-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Name	Type	Rqd/Opt	Description
app-enabled	Boolean	Optional	True indicates that the mobile app is enabled for this console. False indicates that the mobile app is disabled and will not be allowed to access this console.
require-app-password-enabled	Boolean	Optional	True indicates that the mobile app must require its user to set an app password in order to access the app itself. False indicates the user is not required to secure the app with a password.
password-caching-enabled	Boolean	Optional	True indicates that the mobile app is permitted to securely store the user's password on the device for this console. False indicates this is not permitted, and the user must input the password on every logon to this console from the app.
actions-enabled	Boolean	Optional	True indicates that the mobile app is permitted to perform actions against this console and the objects it manages, as permitted by the user's authority. False indicates that no actions are permitted from the mobile app to this console, and the app will be used only for monitoring.
action-settings-activate-partition	action-settings object	Optional	The mobile app settings information for the activate logical partition action.
action-settings-deactivate-partition	action-settings object	Optional	The mobile app settings information for the deactivate logical partition action.
action-settings-start-partition	action-settings object	Optional	The mobile app settings information for the start DPM partition action.
action-settings-stop-partition	action-settings object	Optional	The mobile app settings information for the stop DPM partition action.
action-settings-change-activation-profile	action-settings object	Optional	The mobile app settings information for the change activation profile of a logical partition action.
action-settings-load-os-into-partition	action-settings object	Optional	The mobile app settings information for the load OS into logical partition action.
action-settings-reset-partition	action-settings object	Optional	The mobile app settings information for the reset logical partition action.
action-settings-change-partition-weight	action-settings object	Optional	The mobile app settings information for the change DPM partition or logical partition weights and capping action.
action-settings-change-partition-processors	action-settings object	Optional	The mobile app settings information for the change DPM partition or logical partition processors action.

Name	Type	Rqd/Opt	Description
`action-settings-delete-hardware-message`	action-settings object	Optional	The mobile app settings information for the delete hardware message action.
`action-settings-request-service-hardware-message`	action-settings object	Optional	The mobile app settings information for the request service for a hardware message action.
`action-settings-delete-os-message`	action-settings object	Optional	The mobile app settings information for the delete OS message action.
`action-settings-send-os-command`	action-settings object	Optional	The mobile app settings information for the send OS command action.
`action-settings-manage-remote-firmware-updates`	action-settings object	Optional	The mobile app settings information for the manage remote firmware updates action.
`notifications-enabled`	Boolean	Optional	True indicates that notifications from this console to the mobile app are enabled and will flow to any registered devices. False indicates that notifications will not flow from this console to any device with the mobile app. **Note:** If this is specified as false, then **enhanced-notifications-enabled** must also be false or be unspecified, in which case it will also be implicitly false. [Updated by feature **mobile-enhanced-push**]
`enhanced-notifications-enabled`	Boolean	Optional	True indicates that enhanced notifications from this console to the mobile app are enabled and may flow to any registered devices. False indicates that enhanced notifications will not flow from this console to any device with the mobile app. **Note:** If this is specified as true, then **notifications-enabled** must also be true or be unspecified, in which case it will also be implicitly true. [Added by feature **mobile-enhanced-push**]

Description

This operation sets one or more mobile app preferences for this console.

On successful execution, the preferences passed in the request body have been set on this console, and HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/Task permission to the **HMC Mobile Settings** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	337	One or more of the User URIs specified in an **action-settings** object in the request body is of **type pattern-based**.
403 (Forbidden)	1	The user does not have access to the **HMC Mobile Settings** task.
404 (Not Found)	2	One or more of the User or User Template URIs specified in the **action-settings** object in the request body is not known to this console, or the user does not have permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/set-mobile-app-preferences HTTP/1.1
x-api-session: 5aekjhn8t6g2xlc3xery2flewwey1xwxlnsm0ixhn00p81i1da
content-type: application/json
content-length: 81
{
   "actions-enabled":true,
   "app-enabled":true,
   "password-caching-enabled":false
}
```

Figure 440. Set Mobile App Preferences: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 18 Nov 2016 20:40:53 GMT

<No response body>
```

Figure 441. Set Mobile App Preferences: Response

Get CPC Notification Preferences for Device

The `Get CPC Notification Preferences for Device` operation retrieves the configured mobile app notification preferences for a CPC to a mobile device.

HTTP method and URI

```
POST /api/console/operations/get-device-cpc-notification-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`device-id`	String	Required	The unique identifier of the mobile device to which the preferences being retrieved belong.
`cpc-uri`	String/ URI	Required	The **object-uri** property of the CPC for which the notification preferences apply.

Response body contents

The response body is a JSON object with the following fields:

Name	Type	Description
`new-hardware-message`	Boolean	True indicates that this console is configured to notify the identified device of new hardware messages for the identified CPC. False indicates the device with not be notified. The default value is **false** until modified with the `Update CPC Notification Preferences for Device` operation.
`cpc-acceptable-status-change`	Boolean	True indicates that this console is configured to notify the identified device when the **has-unacceptable-status** property changes for the identified CPC. False indicates the device will not be notified. The default value is **false** until modified with the `Update CPC Notification Preferences for Device` operation.
`new-os-message`	Array of String/ URI	Array of **object-uri** values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which new operating system message events should notify the identified device. Any new OS message on the identified partitions will cause the identified device to be notified. To limit which OS messages lead to notifications, refer to **new-os-message-filtered**. Note that this list of partitions is mutually exclusive with those identified in **new-os-message-filtered**. The default value is empty until modified with the `Update CPC Notification Preferences for Device` operation.
`new-os-message-filtered`	Array of partition-os-message-filter objects	Array of **partition-os-message-filter** objects defining the Partitions (for DPM) or Logical Partitions (for non-DPM) and the filter conditions for which new operating system events should notify the identified device. As opposed to the **new-os-message** property, which notifies the device of all OS messages on an identified partition, this allows notification for only OS messages that pass the filter conditions. Note that this list of partitions is mutually exclusive with those listed in **new-os-message**. The default value is empty until modified with the `Update CPC Notification Preferences for Device` operation.

Name	Type	Description
`partition-acceptable-status-change`	Array of String/ URI	Array of **object-uri** values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which the **has-unacceptable-status** property changes should notify the identified device. The default value is empty until modified with the `Update CPC Notification Preferences for Device` operation.

The partition-os-message-filter nested object contains the following fields:

Name	Type	Description
`partition-uri`	String/ URI	The **object-uri** of the Partition (for DPM) or Logical Partition (for non-DPM) object for which new operating system message events should notify the identified device when they pass the defined filters.
`filters`	Array of os-message-filter objects	Array of **os-message-filter** objects to be applied to new operating system messages on the identified partition. Each filter is considered individually against new OS messages, but all conditions specified within each **os-message-filter** must be met in order for the filter to be considered true. In other words, the logical operation for evaluating a single filter object's conditions is AND, whereas all filter objects are evaluated together with OR.

The os-message-filter nested object contains the following fields:

Name	Type	Description
`priority`	Boolean	Filter to select messages based on their priority. True indicates that a new operating system message passes this filter if its **is-priority** property is set to true. False indicates that it passes this filter if **is-priority** is false. This field is omitted if priority is not to be considered by this filter.
`held`	Boolean	Filter to select messages based on whether they require a response. True indicates that a new operating system message passes this filter if its **is-held** property is set to true. False indicates that it passes this filter if **is-held** is false. This field is omitted if a response requirement is not to be considered by this filter.
`message-text`	String	Filter to select messages based on their message text in order to be notified when an OS emits specific matching messages. It must be a valid Java regular expression. To pass this filter, the string regular expression must match the operating system message's **message-text** property. This field is omitted if message text is not to be considered by this filter.

Description

This operation retrieves the mobile app notification preferences of a CPC for a device from this console. This specifies what notifications are currently being delivered to the mobile app on the identified device for the CPC. Note that a **device-id** may be passed for which an `Update CPC Notification`

`Preferences for Device` operation has not been performed. In this case, default values will be returned, as defined in the "Response body contents" on page 857.

On successful execution, the preference properties are provided in the response body, and HTTP status code 200 (OK) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 857.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	2	The requested **cpc-uri** is not known to this console.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request body is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/get-device-cpc-notification-preferences HTTP/1.1
x-api-session: 64rvm5qub8e2viadxuitl1xfustyhz4wl9j2pcebb1hbyfe0pm
content-type: application/json
content-length: 120
{
   "cpc-uri":"/api/cpcs/9724bf69-038b-3152-841a-3ceb8ee21515",
   "device-id":"dXgLFgPZGMB06nPRB03nFinn17SW:ugACKp8kZfDgR"
}
```

Figure 442. Get CPC Notification Preferences for Device: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 07 Dec 2017 15:21:52 GMT
content-type: application/json;charset=UTF-8
content-length: 467
{
   "cpc-acceptable-status-change":false,
   "new-hardware-message":false,
   "new-os-message":[
      "/api/logical-partitions/1b7d74fd-6de7-3642-93a6-6c3a1a7488b6"
   ],
   "new-os-message-filtered":[
      {
         "filters":[
            {
               "held":true
            },
            {
               "priority":true
            }
         ],
         "partition-uri":"/api/logical-partitions/26d3ade6-3cc9-3841-bcc0-f9f2d8b46f0a"
      }
   ],
   "partition-acceptable-status-change":[
      "/api/logical-partitions/1b7d74fd-6de7-3642-93a6-6c3a1a7488b6",
      "/api/logical-partitions/26d3ade6-3cc9-3841-bcc0-f9f2d8b46f0a"
   ]
}
```

Figure 443. Get CPC Notification Preferences for Device: Response

Update CPC Notification Preferences for Device

The `Update CPC Notification Preferences for Device` operation updates the configured mobile app notification preferences for a CPC to a mobile device.

HTTP method and URI

```
POST /api/console/operations/update-device-cpc-notification-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`device-id`	String	Required	The unique identifier of the mobile device to which the preferences being updated belong.
`cpc-uri`	String/ URI	Required	The **object-uri** property of the CPC for which the notification preferences apply.
`new-hardware-message`	Boolean	Optional	True indicates that this console will now be configured to notify the identified device of new hardware messages for the identified CPC. False indicates the device will not be notified. Note that passing true requires object-access permission to the identified CPC.

Name	Type	Rqd/Opt	Description
`cpc-acceptable-status-change`	Boolean	Optional	True indicates that this console will now be configured to notify the identified device of **has-unacceptable-status** property changes for the identified CPC. False indicates the device will not be notified. Not that passing true requires object-access permission to the identified CPC.
`new-os-message`	Array of String/ URI	Optional	Array of **object-uri** values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which new operating system message events will notify the identified device. Any new OS messages on the identified partitions will cause the identified device to be notified. To limit which OS messages lead to notifications, refer to **new-os-message-filtered**. Note that this list of partitions is mutually exclusive with those identified in **new-os-message-filtered**. Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed partition object must have the parent identified by the **cpc-uri**.
`new-os-message-filtered`	Array of partition-os-message-filter objects	Optional	Array of **partition-os-message-filter** objects defining the Partitions (for DPM) or Logical Partitions (for non-DPM) and the filter conditions for which new operating system events will notify the identified device. As opposed to the **new-os-message** property which notifies the device of all OS messages on an identified partition, this allows notification for only OS messages that pass the filter conditions. Note that this list of partitions is mutually exclusive with those identified in **new-os-message**. Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed Partition object must have the parent identified by the **cpc-uri**.
`partition-acceptable-status-change`	Array of String/ URI	Optional	Array of **object-uri** values of Partition (for DPM) or Logical Partition (for non-DPM) objects for which **has-unacceptable-status** property changes will notify the identified device. Note that object-access permission is required for each passed object, but not necessarily for the identified parent CPC. Additionally, each passed partition object must have the parent identified by the **cpc-uri**.

The partition-os-message-filter nested object contains the following fields:

Name	Type	Rqd/Opt	Description
`partition-uri`	String/ URI	Required	The **object-uri** of the Partition (for DPM) or Logical Partition (for non-DPM) object for which new operating system message events should notify the identified device when they pass the defined filters.
`filters`	Array of os-message-filter objects	Required	Array of **os-message-filter** objects to be applied to new operating system messages on the identified partition. Each filter is considered individually against new OS messages, but all conditions specified within each **os-message-filter** must be met in order for the filter to be considered true. In other words, the logical operation for evaluating a single filter object's conditions is AND, whereas all filter objects are evaluated together with OR. Note that the array must be non-empty.

The os-message-filter nested object contains the following fields:

Name	Type	Rqd/Opt	Description
`priority`	Boolean	Optional	Filter to select messages based on their priority. True indicates that a new operating system message passes this filter if its **is-priority** property is set to true. False indicates that it passes this filter if **is-priority** is false. Omit this field if priority is not to be considered by this filter. Note that at least one filter field is required.
`held`	Boolean	Optional	Filter to select messages based on whether they require a response. True indicates that a new operating system message passes this filter if its **is-held** property is set to true. False indicates that it passes this filter if **is-held** is false. Omit this field if a response requirement is not to be considered by this filter. Note that at least one filter field is required.
`message-text`	String	Optional	Filter to select messages based on their message text in order to be notified when an OS emits specific matching messages. It must be a valid Java regular expression. To pass this filter, the string regular expression must match the operating system message's **message-text** property. Omit this field if message text is not to be considered by this filter. If specified, the string must be non-empty. Note that at least one filter field is required.

Description

This operation updates the mobile app notification preferences of a CPC for a device from this console. This specifies what notifications are to be delivered to the mobile app on the identified device for the CPC.

Note that object-access permission to the CPC is required only to turn on the **new-hardware-message** or **cpc-acceptable-status-change** notification properties. For any **object-uri** included in the **new-os-**

message array or **partition-acceptable-status-change** array, only object-access permission to that partition or logical partition is required.

On successful execution, the notification preference properties are updated for the identified device and CPC from this console, and HTTP status code 204 (No Content) is returned.

Authorization requirements

Object-access permission is required to the CPC to enable notifications for its hardware messages and acceptable status changes. Object-access permission is required to a logical partition or partition to enable notifications for its operating system messages or acceptable status changes.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	2	The requested **cpc-uri** is not known to this console, or one of the CPC-specific notification preferences was asked to be enabled but the user does not have object-access permission to the CPC.
	335	One or more of the requested logical partition or partition URIs is not known to the identified CPC and/or to this console, or the user does not have object-access permission to it.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request body is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/update-device-cpc-notification-preferences HTTP/1.1
x-api-session: 2u80csmk000d1mjpgevfavmhodr71eb2ilocsvbvw6weng0ma8
content-type: application/json
content-length: 318
{
   "cpc-acceptable-status-change":false,
   "cpc-uri":"/api/cpcs/ded1343f-c248-3fc0-afb4-8e54a2ea9647",
   "device-id":"f99e434d-6a38-3487-b331",
   "new-hardware-message":true,
   "new-os-message":[
      "/api/logical-partitions/5709de82-63ee-370f-9928-3a1331332acc",
      "/api/logical-partitions/5989e5ec-b820-37ae-b5f7-81c476a96885"
   ]
}
```

Figure 444. Update CPC Notification Preferences for Device: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Fri, 18 Nov 2016 21:12:06 GMT

<No response body>
```

Figure 445. Update CPC Notification Preferences for Device: Response

List Remote Firmware Updates of the Console

The `List Remote Firmware Updates of the Console` operation returns a list of the remote firmware update operations on a Console.

HTTP method and URI

```
GET /api/console/remote-firmware-updates
```

Query Parameters

Name	Type	Rqd/Opt	Description
`state`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **state** property. Value must be a valid Remote Firmware Update element object **state** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`remote-firmware-updates`	Array of remote-firmware-update-info objects	A list of the remote firmware update operations scheduled on the console. Each element in the list is a remote-firmware-update-info nested object defined in Table 448 on page 864.
`remote-firmware-update-tokens`	Array of remote-firmware-update-token-info objects	A list of remote firmware update token information. Each element in the list is a remote-firmware-update-token-info nested object defined in Table 449 on page 865.

Each nested remote-firmware-update-info object contains the following fields:

Table 448. List Remote Firmware Updates of the Console: remote-firmware-update-info objects

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path (**element-uri**) of the Remote Firmware Update element object.

Table 448. List Remote Firmware Updates of the Console: remote-firmware-update-info objects (continued)		
Field name	**Type**	**Description**
`scheduled-execution-time`	Timestamp	The **scheduled-execution-time** property of the Remote Firmware Update element object.
`target-bundle`	String	The **target-bundle** property of the Remote Firmware Update element object.
`state`	String Enum	The **state** property of the Remote Firmware Update element object.

Each nested remote-firmware-update-token-info object contains the following fields:

Table 449. List Remote Firmware Updates of the Console: remote-firmware-update-token-info objects		
Field name	**Type**	**Description**
`authorization-token`	String (6-8)	The authorization token value
`expiration-date`	Timestamp	The date and time at which this token expires and can no longer be used to create a new remote firmware update.

Description

The `List Remote Firmware Updates of the Console` operation returns a list of the remote firmware update operations that are scheduled to run at a future time on a Console, or were scheduled and are currently running. The Remote Firmware Update Console Element URI, scheduled execution start time, bundle level and current state are returned for each.

The operation also returns a list of information about the authorization tokens that are currently defined on the Console. The authorization token value and expiration date are returned for each.

See "Authorize Remote Firmware Updates" on page 870 for a description of the complete remote firmware update process.

If the **state** query parameter is specified, it is validated to ensure it is a valid value for the Remote Firmware Update Console Element **state** property. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those remote firmware updates that have a matching **state** property. If the **state** parameter is omitted, this filtering is not done.

If no remote firmware updates are to be included in the results due to filtering or lack of any remote firmware updates, an empty list is provided and the operation completes successfully. If no remote firmware update tokens exist, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Remote Firmware Updates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 864.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 450. List Remote Firmware Updates of the Console: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines and invalid value.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Remote Firmware Updates** task.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/remote-firmware-updates HTTP/1.1
x-api-session: 3ay4lwilrq0obc6cpxoi2449ddfry3fr2k99xmm41qqo5ut221
```

Figure 446. List Remote Firmware Updates of the Console: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:51:07 GMT
content-type: application/json;charset=UTF-8
content-length: 547
{
   "remote-firmware-update-tokens":[
      {
         "authorization-token":"736466C5",
         "expiration-date":1614294535719
      },
      {
         "authorization-token":"7993D9ED",
         "expiration-date":1614303108266
      }
   ],
   "remote-firmware-updates":[
      {
         "element-uri":"/api/console/remote-firmware-updates/de52dc87-adb4-4cba-a8c7-
           cb1c4516f495",
         "scheduled-execution-time":1614924900000,
         "state":"scheduled",
         "target-bundle":"H33"
      },
      {
         "element-uri":"/api/console/remote-firmware-updates/97add8d4-d1de-4f91-
           aa80-89e5d5ca5b50",
         "scheduled-execution-time":1615011300000,
         "state":"scheduled",
         "target-bundle":"H34"
      }
   ]
}
```

Figure 447. List Remote Firmware Updates of the Console: Response

Get Console Remote Firmware Update Properties

The `Get Console Remote Firmware Update Properties` operation retrieves the properties of a single Remote Firmware Update element object.

HTTP method and URI

```
GET /api/console/remote-firmware-updates/{remote-firmware-update-id}
```

In this request, the URI variable *{remote-firmware-update-id}* is the element ID of the Remote Firmware Update object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the Remote Firmware Update Console element object as defined in the "Remote Firmware Update Console element object" on page 809. Field names and data types in the JSON object are the same as the property names and data types defined in the "Data model" on page 801.

Description

Returns the current values for the properties of the remote firmware update element object as defined in "Remote Firmware Update Console element object" on page 809.

See "Authorize Remote Firmware Updates" on page 870 for a description of the complete remote firmware update process.

If the API user does not have action/task permission to the **Manage Remote Firmware Updates** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{remote-firmware-update-id}* does not identify a Remote Firmware Update element object on the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Remote Firmware Updates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 867.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 451. Get Console Remote Firmware Update Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Remote Firmware Updates** task.
404 (Not Found)	5	A remote firmware update operation with **element-id** *{remote-firmware-update-id}* does not exist in the Console.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/remote-firmware-updates/de52dc87-adb4-4cba-a8c7-cb1c4516f495 HTTP/1.1
x-api-session: 1mxapd2skpb1pzv6ow9oan6zpus4veno8w81v23s74louuaut3
```

Figure 448. Get Console Remote Firmware Update Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:52:00 GMT
content-type: application/json;charset=UTF-8
content-length: 496
{
   "backup-location":"usb",
   "class":"remote-firmware-update",
   "creation-time":1613757008319,
   "element-id":"de52dc87-adb4-4cba-a8c7-cb1c4516f495",
   "element-uri":"/api/console/remote-firmware-updates/de52dc87-adb4-4cba-a8c7-cb1c4516f495",
   "execution-window":60,
   "parent":"/api/console",
   "scheduled-execution-time":1614924900000,
   "service-contact-email-address":"ssrEmail@example.com",
   "service-contact-name":"ssrName",
   "service-contact-telephone-number":"01233456789",
   "state":"scheduled",
   "target-bundle":"H33"
}
```

Figure 449. Get Console Remote Firmware Update Properties: Response

Delete Console Remote Firmware Update

The `Delete Console Remote Firmware Update` operation deletes a remote firmware update operation scheduled to run at a future time on the Console.

HTTP method and URI

```
DELETE /api/console/remote-firmware-updates/{remote-firmware-update-id}
```

In this request, the URI variable *{remote-firmware-update-id}* is the element ID of the Remote Firmware Update object to be deleted.

Description

The `Delete Console Remote Firmware Update` operation deletes a remote firmware update operation scheduled to run at a future time on the Console.

See "Authorize Remote Firmware Updates" on page 870 for a description of the complete remote firmware update process.

If the API user does not have action/task permission to the **Cancel Scheduled Update** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{remote-firmware-update-id}* does not identify a Remote Firmware Update element object on the Console. A 409 (Conflict) status code is returned if the remote firmware update operation identified by **remote-firmware-update-id** is already running or has already completed. Updated by feature **rcl-progress**]

If the request is valid, the identified remote firmware update is deleted from the Console.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Cancel Scheduled Update** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 452. Delete Console Remote Firmware Update: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Cancel Scheduled Update** task.
404 (Not Found)	5	A remote firmware update operation with **element-id** *{remote-firmware-update-id}* does not exist in the Console.
409 (Conflict)	342	The value of the **state** property of the remote firmware update operation with **element-id** *{remote-firmware-update-id}* is **"running"** and it can therefore no longer be deleted.
	382	The value of the **state** property of the remote firmware update with **element-id** *{remote-firmware-update-id}* indicates that the operation has already completed and it can therefore no longer be deleted. [Added by feature **rcl-progress**]

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/console/remote-firmware-updates/de52dc87-adb4-4cba-a8c7-cb1c4516f495 HTTP/1.1
x-api-session: l5b5w1hg644wcp0kpwcb9mhwj5xyhslge6nrxn8opyc1g2v43
```

Figure 450. Delete Console Remote Firmware Update: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:52:41 GMT

<No response body>
```

Figure 451. Delete Console Remote Firmware Update: Response

Authorize Remote Firmware Updates

The `Authorize Remote Firmware Updates` operation returns a token that allows a service representative to remotely schedule a firmware update.

HTTP method and URI

```
POST /api/console/operations/authorize-remote-firmware-updates
```

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`authorization-token`	String (6-8)	The authorization token that a service representative will need to remotely schedule a firmware update.
`expiration-date`	Timestamp	The date and time at which the token expires and can no longer be used to create a new remote firmware update.

Description

The `Authorize Remote Firmware Updates` operation returns a token that allows a service representative to remotely schedule a firmware update to run at a future time. By invoking this operation and giving the returned token to a service representative, the user is authorizing that service representative to remotely create scheduled operations through a Resource Link® interface that will update the firmware on the Console, or any of the CPCs that it manages. At the scheduled update time, a job will be started on the targeted console or CPC that will update its firmware. The service representative will remotely check the results of that operation and take appropriate actions if the update was not successful.

Authorization tokens are valid for 7 days after they are created. After that point, a service representative will no longer be able to use the token to create new remote firmware updates. Note that the expiration of an authorization token does not affect firmware updates that have already been scheduled using that token. They will run at the scheduled time unless explicitly deleted.

The set of remote firmware update operations that have been created by service representatives can be queried using the `List Remote Firmware Updates of a Console` and `List Remote Firmware Updates of a CPC` operations. Additional details about an individual operation can be retrieved using the `Get Console Remote Firmware Update Properties` and `Get CPC Remote Firmware Update Properties` operations. Individual scheduled operations can be deleted using the `Delete Console Remote Firmware Update` and `Delete CPC Remote Firmware Update` operations.

If the API user does not have action/task permission to the **Generate Token** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) is returned if a Support System connection is not configured on the target console.

If the request is valid, a new authorization token is generated and returned as the value of the **authorization-token** field in the response body.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Generate Token** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 870.

Otherwise, the following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 453. Authorize Remote Firmware Updates: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Generate Token** task.
409 (Conflict)	341	The Console is not configured to connect to the Support System.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/authorize-remote-firmware-updates HTTP/1.1
x-api-session: 3fn9aomdikpn9syqy341o3unliwky0bmoj75u8xpxgdqoulg1i
content-type: application/json
```

Figure 452. Authorize Remote Firmware Updates: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:53:43 GMT
content-type: application/json;charset=UTF-8
content-length: 66
{
   "authorization-token":"16E7DF17",
   "expiration-date":1614362024297
}
```

Figure 453. Authorize Remote Firmware Updates: Response

Update Welcome Text

The `Update Welcome Text` operation updates the welcome message properties of the console.

HTTP method and URI

```
POST /api/console/operations/update-welcome-text
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`welcome-text`	String (1-8192)	Optional	The value that should be assigned to the console's welcome text. If a null value is provided the welcome text is cleared.

Name	Type	Rqd/Opt	Description
`classification-text`	String (1-1024)	Optional	The value that should be assigned to the console's classification text. If a null value is provided the classification text is cleared.
`reflow-welcome-text`	Boolean	Optional	Indicates whether to format the welcome text to fit the width of the user's browser window. If the welcome text is not set this value is ignored.

Description

This operation allows the API user to update or remove the values assigned to the classification and welcome text values within the console. If the user does not have action/task permission to the **Create Welcome Text** task status code 403 (Forbidden) is returned.

The request is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

The request body does not need to specify a value for all properties, but rather can and should contain fields only for the properties to be updated. Object properties for omitted fields remain unchanged by this operation. All fields are considered optional but at least one field must be provided on the request; otherwise, status code 400 (Bad Request) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Create Welcome Text** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated operation-specific errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Create Welcome Text** task.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/operations/update-welcome-text HTTP/1.1
x-api-session:  2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 101
{
   "welcome-text": "Update the welcome text",
   "classification-text": null,
   "reflow-welcome-text": true
}
```

Figure 454. Update Welcome Text: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 16 Jun 2021 15:51:46 GMT
```

Figure 455. Update Welcome Text: Response

Get Console Notification Preferences for Device

The `Get Console Notification Preferences for Device` operation retrieves the configured mobile app notification preferences for the console to a mobile device.

HTTP method and URI

```
POST /api/console/operations/get-device-console-notification-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Name	Type	Rqd/Opt	Description
`device-id`	String	Required	The unique identifier of the mobile device to which the preferences being retrieved belong.

Response body contents

The request body is a JSON object with the following field:

Name	Type	Description
`new-hardware-message`	Boolean	True indicates that this console is configured to notify the identified device of new hardware messages for the console. False indicates the device will not be notified. The default value is **false** until modified with the `Update Console Notification Preferences for Device` operation.

Description

This operation retrieves the mobile app notification preferences of the console for a device from this console. This specifies what notifications are currently being delivered to the mobile app on the identified device for the console. Note that a device-id may be passed for which an `Update Console`

`Notification Preferences for Device` operation has not been performed. In this case, default values will be returned, as defined in the "Response body contents" on page 873.

On successful execution, the preference properties are provided in the response body, and HTTP status code 200 (OK) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 873.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 454. Get Console Notification Preferences for Device: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/get-device-console-notification-preferences HTTP/1.1
x-api-session: p90r9aprjsdjk5wnw66e71eaceyi8ombcuo68i5xj8njwtu2t
content-type: application/json
content-length: 58
{
   "device-id":"dXgLFgPZGMB06nPRB03nFinn17SWugACKp8kZfDgR"
}
```

Figure 456. Get Console Notification Preferences for Device: Request

```
200
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 04 Mar 2022 22:56:35 GMT
content-type: application/json;charset=UTF-8
content-length: 29
{
   "new-hardware-message":true
}
```

Figure 457. Get Console Notification Preferences for Device: Response

Update Console Notification Preferences for Device

The `Update Console Notification Preferences for Device` operation updates the configured mobile app notification preferences for the console to a mobile device.

HTTP method and URI

```
POST /api/console/operations/update-device-console-notification-preferences
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`device-id`	String	Required	The unique identifier of the mobile device to which the preferences being updated belong.
`new-hardware-message`	Boolean	Optional	**True** indicates that this console will now be configured to notify the identified device of new hardware messages for the console. **False** indicates the device will not be notified.

Description

This operation updates the mobile app notification preferences of the console for a device from this console. This specifies what notifications are to be delivered to the mobile app on the identified device for the console.

On successful execution, the notification preference properties are updated for the identified device from this console, and HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 455. Update Console Notification Preferences for Device: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/update-device-console-notification-preferences HTTP/1.1
x-api-session: 1h4ck52vtigdyjdnwpre4ycd6zkyjqkvhnotug2x0pecwsor9j
content-type: application/json
content-length: 88
{
   "device-id":"dXgLFgPZGMB06nPRB03nFinn17SWugACKp8kZfDgR",
   "new-hardware-message":true
}
```

Figure 458. Update Console Notification Preferences for Device: Request

```
204
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 04 Mar 2022 22:54:03 GMT

<No response body>
```

Figure 459. Update Console Notification Preferences for Device: Response

Console Single Step Install

The `Console Single Step Install` operation asynchronously performs a backup of the console firmware and then retrieves, installs, and activates a new bundle of firmware.

HTTP method and URI

```
POST /api/console/operations/single-step-install
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`bundle-level`	String	Optional	Name of the bundle to be installed
`backup-location-type`	String Enum	Optional	Valid values are: • **"ftp"** • **"usb"** Default value: **"usb"**
`accept-firmware`	Boolean	Optional	Accept the previous bundle level before installing the new level. Default value: **true**
`ftp-retrieve`	Boolean	Optional	Retrieve internal code changes from an FTP server. Default value: **false** [Added by feature **hmc-delete-retrieved-internal-code**]

Name	Type	Req/Opt	Description
`ftp-server-host`	String	Optional	The hostname for the FTP server. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **hmc-delete-retrieved-internal-code**]
`ftp-server-user`	String	Optional	The username for FTP server login. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **hmc-delete-retrieved-internal-code**]
`ftp-server-password`	String	Optional	The password for FTP server login. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **hmc-delete-retrieved-internal-code**]
`ftp-server-directory`	String	Optional	The directory to access on the FTP server. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **hmc-delete-retrieved-internal-code**]
`ftp-server-protocol`	String Enum	Optional	The protocol used to connect to the FTP server. Valid values: • **"ftp"** • **"ftps"** • **"sftp"** **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **hmc-delete-retrieved-internal-code**]

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/URI	URI that may be queried to retrieve status updates or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in the "Job status and reason codes" on page 880. The **job-results** field is **null** when this operation is successful. When it is not successful or partially successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful, or a description of firmware updates that are still pending when the operation was not successful.

Description

The `Console Single Step Install` operation installs a firmware bundle on a console. The **ec-mcl-description** property of the Console object provides information about the firmware levels that are known to the console.

Note that it is not possible to remove firmware updates with this operation. Specifying a **bundle-level** that targets firmware updates that are already installed will result in an error rather than a removal of the firmware down to the specified level.

Note that it is possible for the Console object to be configured as a Hardware Management Appliance (HMA). If this is the case, then additional checks are required before commencement of the single step operation. If the Console object is an HMA, then the operation can only begin if the Console object is hosting an alternate virtual SE. Furthermore, the Console object has to be actively communicating with its peer HMA, or the Console object's alternate virtual SE has to be in service status.

Note that it is not possible to view MCL alerts with this operation. Specifying a **bundle-level** that targets firmware updates that trigger an alert during install and activate will result in an error rather than a display of the alert. If the user wishes to proceed with the operation, they will have to log on to the console via the local graphical user interface (GUI) or a remote web browser, then navigate to the **Single Step Console Internal Code** task and rerun the operation.

The internal code that is installed during this operation can be retrieved from either the remote support system or a specified FTP server. In either case, an attempt will be made to retrieve all available internal code changes. If the **bundle-level** field is specified, then all retrieved internal code changes up to the specified bundle boundary will be installed on the Console. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed.

If the **ftp-retrieve** field is specified as **false,** then all possible internal code changes will be retrieved from the remote support system. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed on the Console. Otherwise, the retrieved internal code changes will be installed up to the specified bundle boundary.

If the **ftp-retrieve** field is specified as **true**, then all possible internal code changes will be retrieved from an FTP server. In order to achieve this, the **ftp-server-host**, **ftp-server-user**, **ftp-server-password**, **ftp-server-directory**, and **ftp-server-protocol** fields must all be included in the initial request. If the retrieval was successful, then the set of internal code changes that will be installed depends on the **bundle-level** field. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed on the Console. Otherwise, the retrieved internal code changes will be installed up to the specified bundle boundary. If an error occurs trying to locate the server with the specified hostname, or if authentication to the server cannot be established with provided user credentials, or if the internal code changes cannot be retrieved from their specified location, then a 409 (Conflict) will be returned.

If the API user does not have action/task permission to the **Single Step Console Internal Code** task, a 403 (Forbidden) status code is returned.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, a 202 (Accepted) response is returned and an asynchronous job is started to install the firmware updates identified by the **bundle-level** request body field. Once started, the asynchronous job performs additional validation of the request body fields. If the **bundle-level** does not exist or is already installed on the console, a 400 (Bad Request) is returned in response to a `Query Job Status` request. A 409 (Conflict) is returned if Change Management is not enabled, or if a connection to the Support System is not available on the console. A 409 (Conflict) is also returned if **backup-location-type** is **"usb"** and a USB storage device is not mounted in the console, or is mounted

but has the wrong label, or if **backup-location-type** is **"ftp"** and an error occurs connecting to the FTP server

If the request body contents are valid, the firmware identified by the **bundle-level** request body field is installed. The install process includes the following steps:

- If the value of the **accept-firmware** is **true**, the firmware currently installed on the console is accepted. Note that once firmware is accepted, it cannot be removed.
- A backup of the console is performed. If the value of the **backup-location-type** field is **"usb"**, the backup data is saved to the USB device that is mounted on the console. If the value of the **backup-location-type** field is **"ftp"**, the backup data is saved to an FTP server. The FTP location and login credentials are the same as was used for the last console backup as defined on the **Configure Backup Settings** user interface task.
- If the value of the **ftp-retrieve** field is **true**, then the uninstalled firmware identified by the **bundle-level** field will be retrieved from an FTP server, which is accessed by using the additional fields related to **ftp-retrieve** that are specified in the request body table.
- The uninstalled firmware identified by the **bundle-level** field is installed.
- The newly installed firmware is activated, which includes rebooting the console.

If an error occurred when installing updates, any updates that were successfully installed are rolled back. If a failure occurs after the firmware is accepted, the firmware remains accepted.

When the asynchronous job competes, the response to a `Query Job Status` request will include a **status** of **"complete"**. If the operation was successful, the completion status will be 204 (No Content).

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note there are only a few interruption points in the firmware install process, so it may be some time before the job is canceled, and after some point, will continue on to completion. The job status and reason codes will indicate whether the job was canceled or ran to completion. If the job is successfully canceled, any steps that were successfully completed will not be rolled back.

Special processing is required by the API client to obtain the results for this asynchronous operation because the console that is the target of the API request will restart, possibly multiple times, before the operation finishes. The API session will be broken during each restart, and as a result, the API client will no longer receive notifications, include the notification that the asynchronous job has completed. Once the client detects that the session is broken, it must periodically attempt to reestablish a session. When a session is established, the client should issue a `Query Job Status` request and check to see if the job has finished. If the job is still in progress, the client can either periodically continue to issue `Query Job Status` requests, or register to receive the job completion notification.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission for the **Single Step Console Internal Code** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 877.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have action/task permission to the **Single Step Console Internal Code** task.
409 (Conflict)	2	The console was busy and the request timed out.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
400 (Bad Request)	354	The bundle identified by **bundle-level** does not exist on the target console.
	355	The bundle identified by **bundle-level** is known on the target console, but has not yet been retrieved.
	356	The bundle identified by **bundle-level** is already installed on the target console.
	357	There are MCLs currently installed on the target console that are beyond the bundle identified by **bundle-level**.
	358	The bundle identified by **bundle-level** has been marked as invalid for install.
	401	There are MCL alerts present for install and activate of the bundle identified by **bundle-level**. Log on to the GUI of the target console, then navigate to the **Single Step Console Internal Code** task and rerun the operation.

Job status code	Job reason code	Description
409 (Conflict)	341	The target console does not have an active connection to the Support System.
	342	Connection to the FTP server specified in the **ftp-server-host** field failed. Verify that the hostname is well-defined and exists.
	343	Authentication to the FTP server failed with the provided values for the **ftp-server-user** and **ftp-server-password** fields.
	344	No host key can be found while attempting to access the FTP server specified in the **ftp-server-host** field via the SFTP protocol. Navigate to the “Manage SSH Keys” task in order to add a key for the host.
	345	Connection to the FTP server specified in the **ftp-server-host** field using the FTPS protocol could not be established due to the host's certificate not being recognized. Navigate to the **Certificate Management Task** to import the certificate
	347	Failure occurred while attempting to retrieve the internal code changes contained in the **ftp-server-directory** field on the FTP server.
	360	The **backup-location-type** field is **"ftp"**, but required info for server login is either invalid of undefined.
	361	The **backup-location-type** field is **"ftp"**, but connection to the FTP server failed.
	362	The **backup-location-type** field is **"usb"**, but the target console could not find a USB storage device with the proper label.
	363	The **backup-location-type** field is **"usb"** but more than one USB storage device was found inserted in the target console. Ensure that there is only one USB labeled for backup and it is located in the back/rear of the target console.
	364	The **backup-location-type** field is **"usb"** but an unexpected error occurred while accessing the backup media. Check that the backup media is connected to the target console and try again.
	365	The target console appears to be a Hardware Management Appliance hosting an alternate SE. In order to proceed with the operation, the target console must be communicating with its peer Hardware Management Console, or the Alternate SE hosted on the target console must be in service status. Ensure that one of these requirements is satisfied and retry the operation.
	366	The target console appears to be a Hardware Management Appliance hosting a primary SE. Log on to the GUI of the target console and navigate to the **Alternate Support Element** task. Then, perform the mirror operation followed by the switch operation so that the target console will host the alternate SE. Then, enable Service Status on the primary SE and retry the operation.

Job status code	Job reason code	Description
409 (Conflict)	367	Change Management is not enabled on the target console.
	399	Failure occurred while attempting to retrieve the internal code changes contained in the **ftp-server-directory** field on the FTP server. It is possible that some of the changes were successfully retrieved. Navigate to the **Change Console Internal Code** task to see if any changes were retrieved.
	400	The host key of the FTP server specified in the **ftp-server-host** field via the SFTP protocol does not match the key known to the console. Navigate to the **Manage SSH Keys** task in order to retrieve the proper host key.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/operations/single-step-install HTTP/1.1
x-api-session: 1g7l5uywkf5xeqxwr29rex2g1hws27nqjnrkj54fy5vbfs7ekp
Content-Type: application/json
Content-Length: 80
{
   "accept-firmware":false,
   "backup-location-type":"usb",
   "bundle-level":"H01"
}
```

Figure 460. Console Single Step Install: Request

```
202
Content-Type: application/json;charset=UTF-8
Content-Length: 60
{
   "job-uri":"/api/jobs/df973ee4-c273-11ec-89d9-fa163e920859"
}
```

Figure 461. Console Single Step Install: Response

Report a Console Problem

The `Report a Console Problem` operation reports and requests service for a problem on an HMC. [Added by feature **report-a-problem**]

HTTP method and URI

```
POST /api/console/operations/report-problem
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`customer-name`	String (0-50)	Optional	Name of the customer. May not contain any double-byte characters or ";". Default: "Unknown"
`customer-phone-number`	String (0-20)	Optional	Phone number of customer. May not contain any double-byte characters or ";". Default: "Unknown"
`problem-description`	String (1-510)	Required	Description of the problem. May not contain any double-byte characters or ";".
`problem-type`	String Enum	Required	Identifies the type of problem. One of: • **"health"** - Report the state of the HMC before applying a maintenance action. • **"hmc"** - Report a problem that occurred on the HMC. • **"test"** - Test whether problems can be reported for the HMC.

Description

The `Report a Console Problem` operation reports a problem for the HMC and requests service to repair it.

Problems are reported to the support system for the HMC. Reporting a problem sends the information provided in the request and the machine information that identifies the console to the service provider.

Automatic service call reporting must be enabled on the HMC via the **Remote Service** task to use this operation. If the HMC does not have automatic service call reporting enabled, a 409 (Conflict) status code is returned.

Upon successful problem creation, a 204 (No Content) status code is returned.

If the API user does not have action/task permission to the **Report A Console Problem** task, a 403 (Forbidden) status code is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission for the **Report A Console Problem** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required action/task permissions.

HTTP error status code	Reason code	Description
409 (Conflict)	600	The operation cannot be performed because the HMC does not have automatic service call reporting enabled.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/operations/report-problem HTTP/1.1
x-api-session: 5v1r4vy0gq4tes9uxzel9tvmn59wbhujmt45ko89debv7ubbya
Content-Type: application/json
Content-Length: 134
{
  "customer-name":"Tester",
  "customer-phone-number":"888-888-8888",
  "problem-description":"This is a console problem",
  "problem-type":"hmc"
}
```

Figure 462. Report a Console Problem: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 06 Feb 2023 23:00:39 GMT
<No response body>
```

Figure 463. Report a Console Problem: Response

Console Delete Retrieved Internal Code

The `Console Delete Retrieved Internal Code` operation deletes retrieved internal code that has not been installed on the Console. [Added by feature **hmc-delete-retrieved-internal-code**]

HTTP method and URI

```
POST /api/console/operations/delete-retrieved-internal-code
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`ec-levels`	Array of ec-level objects	Optional	Array of nested **ec-level** objects (defined in Table 519 on page 1157) that indicate the uninstalled engineering change levels to be deleted. Default: All retrieved, uninstalled MCLs are deleted down to the applied levels.

Response body contents

Once the `Console Delete Retrieved Internal Code` request is accepted, the response body contains a JSON object with the following fields

Name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Response body contents" on page 885. The **job-results** field is null when this operation is successful. When it is not successful or partially successful, the **job-results** field contains an object with the following field:

Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The `Console Delete Retrieved Internal Code` operation deletes retrieved internal code that has not been installed on the Console. The firmware is segmented into different subsystems identified by Engineering Change (EC) numbers. Sets of firmware updates within a single EC are packaged together and assigned a Microcode Level (MCL). MCL packages are installed sequentially, so an MCL implies not only the firmware updates that were packaged with that MCL, but all of the MCLs that preceded it in the EC stream. If the **ec-levels** field is present in the request body, it identifies a set of retrieved, but uninstalled EC MCLs that are to be deleted on the Console. If the **ec-levels** field is not present, then all the firmware that is currently retrieved, but not installed, will be deleted from the Console.

If the API user does not have action/task permission to the **Change Console Internal Code** task, a 403 (Forbidden) status code is returned. A 409 (Conflict) status code is returned if the Console is busy. If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, a 202 (Accepted) response is returned and an asynchronous job is started to remove the firmware updates identified by the **ec-levels** request body field. Once started, the asynchronous job performs additional validation of the request body fields. If the **ec-levels** field is present in the request body and references an Engineering Change (EC) number or Microcode Level (MCL) that does not exist on the Console, a 400 (Bad Request) is returned in response to a Query Job Status Request. A 409 (Conflict) is returned if Change Management is not enabled, or if a connection to the Support System is not available on the Console, or if the **ec-levels** field identifies microcode levels that are already installed on the system instead of only being retrieved.

If the request body contents are valid, the firmware updates identified by the **ec-levels** request body field are deleted. If an error occurred when deleting the updates, then only the updates that were unsuccessfully deleted will remain on the system; any updates that were deleted before reaching an error will remain deleted upon completion of the operation.

When the asynchronous job competes, the response to a Query Job Status request will include a status of "complete". If the operation was successful, the completion status will be 204 (No Content).

Authorization requirements

This operation has the following authorization requirements:

- Action/task permission for the **Change Console Internal Code** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Job status and reason codes" on page 886.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Change Console Internal Code** task.
409 (Conflict)	2	The target console was busy and the request timed out.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	The operation completed successfully.
400 (Bad Request)	378	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that does not identify a known component and is therefore not valid.
400 (Bad Request	367	Change Management is not enabled on the target console.
	383	There are no internal code changes on the target console, so the change internal code operation could not be performed.
	385	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that is not properly bounded by the current applied and staged levels.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/operations/delete-retrieved-internal-code HTTP/1.1
x-api-session: 4pse48gtmvnkgmsx9sbccibq4xsrw7e6ekpfy5hkukm4n3d8n
Content-Type: application/json
Content-Length: 51
{
   "ec-levels":[
      {
         "mcl":"001",
         "number":"P30805"
      }
   ]
}
```

Figure 464. Console Delete Retrieved Internal Code: Request

```
202 Accepted
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Wed, 25 Oct 2023 14:25:01 GMT
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/485bbd9e-7342-11ee-bc9e-fa163e440a5c"
}
```

Figure 465. Console Delete Retrieved Internal Code: Response

List Console API Features

The `List Console API Features` operation returns information about the API features available on the console. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/operations/list-features
```

Response body contents

On successful completion, the response body is a JSON array of String values, each of which identifies an available API feature. The order in which these strings are returned is unspecified. The possible feature names are listed in “API features” on page 103.

Description

This operation lists the API features available on the console. Beginning with API version 4.10, API clients must use this operation and the `List CPC API Features` operation to determine if specific new or changed API functionality is available.

If the **name** query parameter is specified, the returned list is limited to those API features that have a **name** field matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of a fully-authenticated API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, and the response body is provided as described in "Response body contents" on page 887.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/list-features HTTP/1.1
x-api-session: 5nihezq2ojefbh3umb5tkdl91gonu6dyrjezv8trwqrwwgr0ti
```

Figure 466. List Console API Features: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 31 Jan 2023 13:51:47 GMT
Content-Type: application/json
Content-Length: 188
[
   "environmental-metrics",
   "cpc-install-and-activate",
   "pmg-child-management-permission",
   "secure-boot-with-certificates",
   "report-a-problem",
   "dpm-smcd-partition-link-management",
   "oem-hmc-ids"
]
```

Figure 467. List Console API Features: Response

Inventory service data

Information about the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Console objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included.

For each Console object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Console Properties" on page 812. That is, the data provided is the same as would be provided if a `Get Console Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single console. This object would appear as one array entry in the response array:

```
{
   "class": "console",
   "description": "Endicott Test HMC",
   "ec-mcl-description": {
      "ec": [
         {
            "description": "Hardware Management Console Framework",
            "mcl": [
               {
                  "last-update": 1422026097000,
                  "level": "39",
                  "type": "retrieved"
               },
               {
                  "last-update": 1423074018000,
                  "level": "39",
                  "type": "activated"
               },
               {
                  "last-update": null,
                  "level": "000",
                  "type": "accepted"
               },
               {
                  "last-update": null,
                  "level": "39",
                  "type": "installable-concurrent"
               },
               {
                  "last-update": null,
                  "level": "1",
                  "type": "removable-concurrent"
               }
            ],
            "number": "N98841",
            "part-number": "00LY737",
            "type": "SYSTEM"
         },
```


Figure 468. Console object: Sample inventory data (Part 1)

```
            {
               "description": "Enablement of new features   ",
               "mcl": [
                  {
                     "last-update": null,
                     "level": "000",
                     "type": "retrieved"
                  },
                  {
                     "last-update": null,
                     "level": "000",
                     "type": "activated"
                  },
                  {
                     "last-update": null,
                     "level": "000",
                     "type": "accepted"
                  },
                  {
                     "last-update": null,
                     "level": "000",
                     "type": "installable-concurrent"
                  },
                  {
                     "last-update": null,
                     "level": "000",
                     "type": "removable-concurrent"
                  }
               ],
               "number": "N98844",
               "part-number": "00LY740",
               "type": "ENABLE1"
            }
         ]
      },
```


Figure 469. Console object: Sample inventory data (Part 2)

```
      "ip-swapping-available": true,
       "is-auto-switch-enabled": true,
       "is-locked": false,
       "machine-info": {
          "machine-model": "PBC",
          "machine-serial": "KQ0N5RF",
          "machine-type": "7382"
       },
       "name": "ZFXHMC2",
       "network-info": {
          "paired-hmc": {
             "hmc-name": "ZFXHMC1",
             "ipv4-address": [
                "192.0.2.0",
                "192.0.3.0"
             ],
             "ipv6-address": [
                "2001:0db8:ffb:1:5ef3:fcff:feaf:deb1",
                "2001:0db8:d89b:1:5ef3:fcff:feaf:deb1",
                "2001:0db8:0:0:5ef3:fcff:feaf:deb1"
             ]
          },
          "this-hmc": [
             {
                "domain-name": "",
                "hmc-name": "ZFXHMC2",
                "interface-name": "eth0",
                "ipv4-address": [
                   {
                      "ip-address": "192.0.2.0",
                      "subnet-mask": "255.255.255.0"
                   }
                ],
                "ipv6-address": [
                   {
                      "ip-address": " 2001:0db8:fffb:1:5ef3:fcff:feae:8019",
                      "prefix-length": 64
                   },
                   {
                      "ip-address": "2001:0db8:d89b:1:5ef3:fcff:feae:8019",
                      "prefix-length": 64
                   },
                   {
                      "ip-address": "2001:0db8:0:0:5ef3:fcff:feae:8019",
                      "prefix-length": 64
                   }
                ],
                "is-private": false,
                "mac": "5CF3FCAE8019"
             },
```


Figure 470. Console object: Sample inventory data (Part 3)

```
            {
                  "domain-name": "",
                  "hmc-name": "ZFXHMC2",
                  "interface-name": "eth1",
                  "ipv4-address": [
                     {
                        "ip-address": "192.0.2.0",
                        "subnet-mask": "255.255.255.0"
                     }
                  ],
                  "ipv6-address": [
                     {
                        "ip-address": "2001:0db8:0:0:5ef3:fcff:feae:801a",
                        "prefix-length": 64
                     }
                  ],
                  "is-private": false,
                  "mac": "5CF3FCAE801A"
               }
            ]
      },
      "object-id": "ec982d6c-bcc1-3ae8-b39c-a2efd14734b4",
      "object-uri": "/api/console",
      "paired-role": "primary",
      "parent": null,
      "version": "2.13.0"
   }
```


Figure 471. Console object: Sample inventory data (Part 4)

User-related-access permission

An HMC user has access to certain information about their own user account. This information is contained in their User object and related objects. An API user's access permission to their own User object and specific related objects is known as user-related-access permission. Through such permission, those objects will be included in a List `<class>` operation, unless otherwise filtered out, and the API user is permitted to issue a Get `<class>` Properties operation on them, unless specifically prohibited.

The object types included in user-related-access permission are:

- User
- User Role
- User Pattern
- Password Rule
- LDAP Server Definition
- MFA Server Definition

User-related-access permission includes the following:

- Permission for an object of the above types to be included in the response body of a List `<class>` operation.
- Permission to view properties of objects of the following types through the `Get <class> Properties` and `Get Inventory` operations:
  - User
  - User Role
  - User Pattern
  - Password Rule
- Permission to update certain properties of the User object. See the "Data model" on page 893 or the `Update User Properties` operation for details.

User object

A User object represents a single Hardware Management Console user. There are different types of console users. A typical customer-defined user is known as a standard user. A user template defines certain attributes of a group of users whose user IDs match the expression in a User Pattern; these definitions are known as template users. When a user logs on with a user ID that matches the expression in a User Pattern, a pattern-based user is created. There are certain user definitions supplied by the system; they are known as system-defined users.

All API users are permitted to see their own User object in a `List Users` response, issue `Get User Properties` for their own User object and, with the exception of pattern-based users, issue `Update User Properties` to alter certain properties of their own User object. An API user with action/task permission to the `Manage Users` task is permitted to view and change any standard or system-defined User object. An API user with action/task permission to the `Manage User Templates` task is permitted to view and change any template User object.

System-defined users

In most respects, system-defined users are indistinguishable from standard users. They can be modified or even deleted. Most properties of system-defined users may be changed, but certain others are immutable; the immutable properties are denoted as such in the data model section that follows. While system-defined users can be deleted and their name reused for a standard user definition, that practice is discouraged due to the likely confusion such a situation would cause. The typical system-defined users include the following:

- ACSADMIN
- ADVANCED
- OPERATOR
- SERVICE
- SYSPROG
- STORAGEADMIN

Data model

This object includes the properties defined in the "Base managed object properties schema" on page 100, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

The following class-specific specializations apply to the other base managed object properties:

Table 456. User object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the User object is of the form /api/users/*{user-id}*, where *{user-id}* is the value of the **object-id** property (not the **name** property) of the User object.
parent	—	String/ URI	The canonical URI path of the Console object.
class	—	String	The class of a User object is **"user"**.

Table 456. User object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(ro)	String	If type is not **"template"**: the name (console user ID) of the User object. It must be 4-320 characters in length and consist only of alphanumeric characters and the following special characters: "**@<+:#'="&*();-/\,%_>.?**". This name must be unique among all users whose type is not **"template"** defined on the console. While preexisting names may contain a double quote (") character, it is not permitted in new user names. If type is **"template"**: the name of the template. While preexisting template names are virtually unrestricted in terms of length and characters, new template names must conform to the length and character requirement of the **name** property described in the "Base managed object properties schema" on page 100. This name must be unique among all template definitions on the console. For the purpose of verifying uniqueness only, this name is treated in a case-insensitive fashion when used to create a new User object of any type.
description	(w)(pc)	String (0-1024)	The description of the User object. Default: an empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Note: Some properties are only valid for users of a specific type. Such properties are only included in the User object if the user is of that type, as indicated by its **type** property. For example, a user with a **type** of **"standard"** includes the **disabled** property but not the **user-pattern-uri** property.

Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **password-rule-uri** is **null** when the **authentication-type** value is **"ldap"**.

Table 457. User object: class specific additional properties			
Name	**Qualifier**	**Type**	**Description**
type	—	String Enum	The type of user. Supported values are: • **"standard"** - a standard, normal user. • **"template"** - a user template. • **"pattern-based"** - a user created dynamically from a User Pattern and its associated template. • **"system-defined"** - a user supplied by the system. Certain properties of system-defined users are immutable.
user-pattern-uri	—	String/ URI	The canonical URI path of the User Pattern object upon which this user is based. Prerequisite: **type** is **"pattern-based"**

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
user-template-uri	—	String/ URI	The canonical URI path of the User Template object upon which this user is based. Prerequisite: **type** is **"pattern-based"**
disabled	(w)(pc)	Boolean	Indicates whether the user is currently disabled. When disabled, the user is prevented from logging on to the console through either the UI or the Web Services APIs. Prerequisite: **type** is not **"template"**. Default: **false**
authentication-type	(w)(pc)	String Enum	The type of user ID and password authentication used for this user, which must be one of the following: • **"local"** - the console performs the authentication. • **"ldap"** - authentication is delegated to the LDAP server identified in the **ldap-server-definition-uri** property. If **type** is **"template"**, this must be **"ldap"**. **Note:** The value of this property is a prerequisite for certain other properties. Changing this value requires certain properties to be included in the same request; see the `Update User Properties` operation for details.
password-rule-uri	(w)(pc)	String/ URI	The canonical URI path of the Password Rule for this user. Prerequisite: **authentication-type** is **"local"**.
password	(wo)(pc)	String	The console logon password for this user. The specific length, character and other requirements on this password are controlled by the authentication type and Password Rule assigned to this user. Note the (wo) qualifier; this field may be altered through an API, but it is not included in the response when this object's properties are retrieved through an API.
password-expires	—	Integer	The time interval, in days, until the user's current password expires. A value of 0 indicates that the password will expire within the next 24 hours. A value of -1 indicates that the HMC does not enforce password expiration for this user; however, if this user is authenticated with an external authentication mechanism (e.g. LDAP) such expiration might be enforced by that mechanism.
force-password-change	(w)(pc)	Boolean	Indicates whether the user should be forced to change their console logon password the next time they log in. Prerequisite: **authentication-type** is **"local"** Default: **true**

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ldap-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the configuration object for the LDAP server used for authentication of this user. Prerequisite: **authentication-type** is **"ldap"**.
userid-on-ldap-server	(w)(pc)	String (0-32)	The user ID for this user on the LDAP server identified in **ldap-server-definition-uri**, or **null** if the user's console user ID (value of the **name** property) should be used. See the LDAP Server Definition object for more information on how this property is used. Prerequisite: **authentication-type** is **"ldap"** and **type** is not **"template"**. Default: an empty string
session-timeout	(w)(pc)	Integer (0-525600)	The session timeout in minutes for this user. This is the interval over which a user's UI session can run before being prompted for identity verification. 0 indicates no timeout. Default: 0
verify-timeout	(w)(pc)	Integer (0-525600)	The verification timeout in minutes for this user. This is the amount of time allowed for the user to re-enter their password after being prompted due to a session timeout (see the **session-timeout** property). 0 indicates no timeout. Default: 15
idle-timeout	(w)(pc)	Integer (0-525600)	The idle timeout in minutes for this user. This is the amount of time the user's UI session can be idle before it is disconnected. 0 indicates no timeout. Default: 0
min-pw-change-time	(w)(pc)	Integer (0-525600)	The minimum password change time in minutes for this user. This is the minimum amount of time that must elapse between changes to this user's password. 0 indicates no minimum; that is, the password can be changed immediately after it has just been changed. Prerequisite: **authentication-type** is **"local"**. Default: 0
max-failed-logins	(w)(pc)	Integer (0-525600)	The maximum number of failed login attempts for this user. This is maximum number of consecutive failed login attempts before the user is temporarily disabled for the amount of time specified in the **disable-delay** property. 0 indicates that the user is never disabled due to failed login attempts. Default: 3

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
disable-delay	(w)(pc)	Integer (0-525600)	The time in minutes that the user is disabled after exceeding the maximum number of failed login attempts specified in the **max-failed-logins** property. 0 indicates that the user is not disabled for any period of time after reaching the maximum number of invalid login attempts. Default: 1
inactivity-timeout	(w)(pc)	Integer (0-525600)	The inactivity timeout in days for this user. This is the maximum number of days of inactivity (consecutive days with no login) before the user is disabled. 0 indicates no timeout. Default: 0
disruptive-pw-required	(w)(pc)	Boolean	Indicates whether the user's password is required to perform disruptive actions through the UI. Default: true
disruptive-text-required	(w)(pc)	Boolean	Indicates whether text input is required to perform disruptive actions through the UI. Default: false
allow-remote-access	(w)(pc)	Boolean	Indicates whether the user is allowed to access the HMC through its remote web server interface Default: false
allow-management-interfaces	(w)(pc)	Boolean	Indicates whether the user is allowed access to management interfaces. This includes access to the Web Services APIs. Default: false
max-web-services-api-sessions	(w)(pc)	Integer (0-9999)	The maximum number of simultaneous Web Services API sessions the user is permitted to have. Default: 100
web-services-api-session-idle-timeout	(w)(pc)	Integer (1-360)	The idle timeout in minutes for Web Services API sessions created by this user. This is the amount of time a Web Services API session can be idle before it is terminated. Default: 360
user-roles	(c)(pc)	Array of String/ URI	The list of user roles defined for this user. Each element in this array is a canonical URI path for a User Role object. The roles provided in this list can change as a result of the `Add User Role to User` and `Remove User Role from User` operations. This property is immutable if **type** is **"system-defined"**.

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
default-group-uri	(w)(pc)	String/ URI	The canonical URI path of the user's default group or **null** if the user has no default group. Managed objects created by this user automatically become members of this group. The user must have object-access permission to this group. This must be a user-defined group to which the user has object-access permission. API users are permitted to change their own default group designation through the `Update User Properties` operation. Default: null
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.
multi-factor-authentication-required	(w)(pc)	Boolean	Indicates whether the user is required to use the HMC's built-in MFA support. If **true**, the user is required to enter their current TOTP multi-factor authentication code (time-based one-time password) in addition to their logon password during UI and API logons. Setting this to **true** will cause **mfa-types** to be set to a one-element array containing **"hmc-totp"**. Setting this to **false** will cause **mfa-types** to be set to **null** if **"hmc-totp"** is present. Default: **false**
force-shared-secret-key-change	(w)(pc)	Boolean	Indicates whether the user is required to establish a new shared secret key during the next logon. The shared secret key is used to calculate the user's current multi-factor authentication code, which is required during logon. Prerequisite: **multi-factor-authentication-required** is **true** Default: **false**
email-address	(w)(pc)	String (0-254)	The user's email address or **null** if the user has no email address. This email address must roughly adhere to Internet Engineering Task Force (IETF) RFC 822.

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
mfa-types	(w)(pc)	Array of String Enum	Identifies the types of multi-factor authentication (MFA) the user is required to use when logging onto the HMC, or **null** if MFA is not required. When setting this property, the API client program is responsible for keeping it and **multi-factor-authentication-required** consistent. Each element of this array must be unique and must be one of the following: • **"hmc-totp"** - Time-based one-time password validated by the HMC. This is equivalent to setting **multi-factor-authentication-required** to **true**. If present in the array, it must be the only element of the array. • **"mfa-server"** - Additional factors validated by an MFA server. Default: **null**
primary-mfa-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the MFA Server Definition object for the primary MFA server used to authenticate the user. Prerequisite: **mfa-types** contains **"mfa-server"** Default: **null**
backup-mfa-server-definition-uri	(w)(pc)	String/ URI	The canonical URI path of the MFA Server Definition object for the backup MFA server used to authenticate the user, or **null** if there is no backup server. Must specify a different MFA server than the primary MFA server. Prerequisite: **mfa-types** contains **"mfa-server"** Default: **null**
mfa-policy	(w)(pc)	String (1-64)	The name of the MFA policy, such as a RACF Policy, that applies to the user when an MFA server authenticates the user. It must identify a policy whose only MFA factor is the RSA SecurID factor. Prerequisite: **mfa-types** contains **"mfa-server"** Default: **null**
mfa-userid	(w)(pc)	String (1-64)	The MFA user ID. This is a user ID, such as a RACF user ID, that identifies this user to the MFA server that authenticates this user. For User objects with a **type** of **"pattern-based"**, this property's default value may be overridden by the LDAP attribute identified by **mfa-userid-override**. Prerequisite: **type** is not **"template"**, and **mfa-types** contains **"mfa-server"** Default: same value as **name** property

Table 457. User object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
mfa-userid-override	(w)(pc)	String (1-256)	The name of the LDAP attribute that contains the MFA user ID, such as a RACF user ID, that identifies the user to the MFA server that authenticates the user, or **null** if there is no such attribute. This can be used to override the value of the **mfa-userid** property during authentication. If the named LDAP attribute does not exist in a user's directory entry, or it exists but is empty, then the user's MFA user ID is not altered. Prerequisite: **type** is **"template"**, and **mfa-types** contains **"mfa-server"** Default: **null**

List Users

The `List Users` operation lists standard, template and system-defined users defined to the console. With one very specific exception, pattern-based users are never included in the response to the `List Users` operation.

HTTP method and URI

```
GET /api/console/users
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid **type** property value, with the exception of **"pattern-based"**.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`users`	Array of objects	Array of nested user-info objects as described in the next table.

Each nested user-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the User object.
`name`	String	The **name** property of the User object.
`type`	String Enum	The **type** property of the User object.

Description

The `List Users` operation lists users defined to the console. Some basic properties are provided for each user.

If the **name** query parameter is specified, the returned list is limited to those users that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid user **type** property value for this operation. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those users that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A user is included in the list only if the API user has sufficient access permission to that object. The access permission requirements for User objects vary depending on the type of User object. All API users have user-related-access permission to their own User object. Action/task permission to the **Manage Users** task includes access permission to all non-template Users, and action/task permission to the **Manage User Templates** task includes access permission to all template users. If there is a User object to which the API user does not have permission, that object is omitted from the list, but no error status code results.

A pattern-based user is only included in the response if it meets the filtering criteria and is the User object for the API user.

If there are no users defined to the console or if no users are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User object included in the response body, or, depending on the type of User object, action/task permission to the **Manage Users** task or the **Manage User Templates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 900.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 458. List Users: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/users?type=system-defined&name=.*ADMIN HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 472. List Users: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:30 GMT
content-type: application/json;charset=UTF-8
content-length: 229
{
   "users":[
      {
         "name":"ACSADMIN",
         "object-uri":"/api/users/ae8aed68-3dc0-11e4-8dd1-1c6f65065a91",
         "type":"system-defined"
      },
      {
         "name":"STORAGEADMIN",
         "object-uri":"/api/users/ae6bf048-3dc0-11e4-8dd1-1c6f65065a91",
         "type":"system-defined"
      }
   ]
}
```

Figure 473. List Users: Response

Get User Properties

The `Get User Properties` operation retrieves the properties of a single User object that is designated by its object ID. With one very specific exception, this operation does not support pattern-based users.

HTTP method and URI

```
GET /api/users/{user-id}
```

In this request, the URI variable *{user-id}* is either the object ID of the User object whose properties are to be returned or the keyword value **"this-user"** which designates the API user that issued the request.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User object as defined in the data model section. Field names and data types in the JSON object are the same as the property names and data types defined in the "Data model" on page 893.

Description

This operation returns the current properties of a single User object that is designated by *{user-id}*.

On successful execution, all of the current properties as defined in the data model for the User object, except those designated as write-only properties, are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User object and the API user must have access permission to it. All API users have user-related-access permission to their own User object. Action/task permission to the **Manage Users** task includes access permission to all non-template Users, and action/task permission to

the **Manage User Templates** task includes access permission to all template users. If the URI path does not designate an existing User object or the API user does not have access permission to it, status code 404 (Not Found) is returned.

This operation does not support pattern-based users, unless the target of the operation is the User object for the API user that issued the request.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User object specified in the request URI, or, depending on the type of User object specified in the request URI, action/task permission to the **Manage Users** task or the **Manage User Templates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 902.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 474. Get User Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Aug 2018 21:27:30 GMT
content-type: application/json;charset=UTF-8
content-length: 1387
{
   "allow-management-interfaces":false,
   "allow-remote-access":false,
   "authentication-type":"local",
   "backup-mfa-server-definition-uri":null,
   "class":"user",
   "default-group-uri":null,
   "description":"Gabby McRosie - company president",
   "disable-delay":1,
   "disabled":false,
   "disruptive-pw-required":true,
   "disruptive-text-required":false,
   "email-address":"finn@example.com",
   "force-password-change":true,
   "force-shared-secret-key-change":false,
   "idle-timeout":0,
   "inactivity-timeout":0,
   "is-locked":false,
   "ldap-server-definition-uri":null,
   "max-failed-logins":3,
   "max-web-services-api-sessions":100,
   "mfa-policy":null,
   "mfa-types":[
     "hmc-totp"
    ],
   "mfa-userid":null,
   "mfa-userid-override":null,
   "min-pw-change-time":0,
   "multi-factor-authentication-required":true,
   "name":"Gabby",
   "object-id":"e9e8d20a-4a7a-11e4-91ee-1c6f65065a91",
   "object-uri":"/api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91",
   "parent":"/api/console",
   "password-expires":-1,
   "password-rule-uri":"/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
   "primary-mfa-server-definition-uri":null,
   "replication-overwrite-possible":false,
   "session-timeout":0,
   "type":"standard",
   "user-roles":[
      "/api/user-roles/ea6f9b14-4a7a-11e4-affa-1c6f65065a91",
      "/api/user-roles/ea41a664-4a7a-11e4-91ee-1c6f65065a91",
      "/api/user-roles/ea094df0-4a7a-11e4-8777-1c6f65065a91"
   ],
   "userid-on-ldap-server":null,
   "verify-timeout":15,
   "web-services-api-session-idle-timeout":360
}
```

Figure 475. Get User Properties: Response

Update User Properties

The `Update User Properties` operation updates the properties of a single User object that is designated by its object ID. This operation is not valid for pattern-based users.

HTTP method and URI

```
POST /api/users/{user-id}
```

In this request, the URI variable *{user-id}* is the object ID of the User object whose properties are to be updated or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the User object specified by *{user-id}*.

The URI path must designate an existing User object and the API user must have permission to update it. All API users have user-related-access permission to their own User object, which includes permission to update certain properties. Action/task permission to the **Manage Users** task includes update permission to all non-template users, and action/task permission to the **Manage User Templates** task includes update permission to all template users. If the URI path does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user does not have action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, an attempt to update any field other than the user's own **password** or **default-group-uri** field results in status code 403 (Forbidden), and specifying a group to which the API user does not have object-access permission in **default-group-uri** results in status code 404 (Not Found).

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) or 409 (Conflict) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., attempting to set **password-rule-uri** when the **authentication-type** value is **"ldap"**). An attempt to update a pattern-based user is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **authentication-type** is changed from **"ldap"** to **"local"**, and **force-password-change** is not defined in the request body, **force-password-change** will be defaulted to false). Note however that certain fields must be included in the request body if the request alters the **authentication-type** property, because they are required in order to perform the newly specified type of logon authentication. Specifically, changing **authentication-type** to **"local"** requires that the following properties also be specified in the same request: **password**, **password-rule-uri**. Changing **authentication-type** to **"ldap"** requires that the following property also be specified: **ldap-server-definition-uri**. Also, certain fields must be included in the request body if the request alters, directly or indirectly, the **mfa-types** property, because they are required in order to perform the newly specified type of multi-factor authentication. Specifically, adding **"mfa-server"** to **mfa-types** requires that the following properties also be specified in the same request: **primary-mfa-server-definition-uri**, **mfa-policy**.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- For a user to update their own **password** or **default-group-uri** property, user-related-access permission to the User object specified in the request URI or action/task permission to the **Manage Users** task is required.

- An API user with action/task permission to the **Manage Users** task or the **Manage User Templates** task, depending on the type of User object specified in the request URI, may update any writable property of that User object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 459. Update User Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	311	The new password does not conform to the requirements of the password policy in effect for this user.
	314	This operation is not supported for an object of this type. Pattern-based users may not be updated.
	330	The operation is not valid because the API user cannot disable their own user ID.
	337	The **primary-mfa-server-definition-uri** in the request body and the **backup-mfa-server-definition-uri** in the request body designate the same MFA Server Definition object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.
	323	The **password-rule-uri** field in the request body does not designate an existing Password Rule object.
	324	The **ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	325	The **default-group-uri** field in the request body does not designate an existing resource of the expected type, or designates a resource for which the user identified by the request URI does not have object-access permission.
	338	The **primary-mfa-server-definition-uri** in the request body does not designate an existing MFA Server Definition object.
	339	The **backup-mfa-server-definiton-uri** in the request body does not designate an existing MFA Server Definition object.

Table 459. Update User Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	8	The request body contains a field whose presence or value is inconsistent with the current state of the User object.
	321	The user's **authentication-type** property cannot be changed at this time. The specified user is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.
	337	The operation could not be completed because it would result in the **primary-mfa-server-definition-uri** and the **backup-mfa-server-definition-uri** designating the same MFA Server Definition object.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 145
{
   "allow-management-interfaces":true,
   "description":"A new and improved description of this User",
   "web-services-api-session-idle-timeout":240
}
```

Figure 476. Update User Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:30 GMT

<No response body>
```

Figure 477. Update User Properties: Response

Add User Role to User

The Add `User Role to User` operation adds a specified User Role to a specified user. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

```
POST /api/users/{user-id}/operations/add-user-role
```

In this request, the URI variable *{user-id}* is the object ID of the user to which a User Role is to be added or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`user-role-uri`	String/ URI	Required	The canonical URI path of the User Role to be added.

Description

This operation adds a User Role to a user.

On successful execution of this operation the User Role specified in the request body has been added to the user identified in the request URI.

The request body is validated against the schema described in "Request body contents" on page 907. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. If the request body does not designate an existing User Role, status code 404 (Not Found) is returned. If the specified object is already in the collection of the user's User Roles, status code 409 (Conflict) is returned. An attempt to update the User Role collection of a system-defined or pattern-based user is not valid and fails with status code 400 (Bad Request).

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to modify a standard user or the **Manage User Templates** task to modify a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 460. Add User Role to User: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. The User Role collection of system-defined and pattern-based users may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.

Table 460. Add User Role to User: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
409 (Conflict)	315	The object designated by the URI in the request body is already in the user's collection of User Roles.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91/operations/add-user-role HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 73
{
   "user-role-uri":"/api/user-roles/eaecdf34-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 478. Add User Role to User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 479. Add User Role to User: Response

Remove User Role from User

The `Remove User Role from User` operation removes a specified User Role from a specified user. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

```
POST /api/users/{user-id}/operations/remove-user-role
```

In this request, the URI variable *{user-id}* is the object ID of the user from which a User Role is to be removed or the special keyword value **"this-user"** which designates the API user that issued the request.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`user-role-uri`	String/ URI	Required	The canonical URI path of the User Role to be removed.

Description

This operation removes a User Role from a user.

On successful execution of this operation the User Role specified in the request body has been removed from the user identified in the request URI.

The request body is validated against the schema described in "Request body contents" on page 909. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User object, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. If the specified object is not in the collection of the user's User Roles, status code 409 (Conflict) is returned. An attempt to update the User Role collection of a system-defined or pattern-based user is not valid and fails with status code 400 (Bad Request).

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to modify a standard user or the **Manage User Templates** task to modify a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 461. Remove User Role from User: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. The User Role collection of system-defined and pattern-based users may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	316	The object designated by the URI in the request body is not in the user's collection of User Roles.
	321	The User Role cannot be removed at this time. The user is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.
	328	The User Role cannot be removed at this time, because doing so would leave the user without object-access permission to their default group.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91/operations/remove-user-role HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 73
{
   "user-role-uri":"/api/user-roles/eaecdf34-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 480. Remove User Role from User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 481. Remove User Role from User: Response

Create User

The `Create User` operation creates a standard or template User object with the given properties. This operation is not valid for system-defined or pattern-based users.

HTTP method and URI

```
POST /api/console/users
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the user's **name** property. Note that the length and character requirements that apply to this field are dependent on the value of the **type** field.
`description`	String	Optional	The value to be set as the user's **description** property.
`type`	String Enum	Required	The value to be set as the user's **type** property. Must be **"standard"** or **"template"**.
`disabled`	Boolean	Optional	The value to be set as the user's **disabled** property.
`authentication-type`	String Enum	Required	The value to be set as the user's **authentication-type** property.
`password-rule-uri`	String/ URI	Required if **authentication-type** is **"local"**	The value to be set as the user's **password-rule-uri** property.

Field name	Type	Rqd/Opt	Description
password	String	Required if **authentication-type** is **"local"**	The value to be set as the user's **password** property.
force-password-change	Boolean	Optional	The value to be set as the user's **force-password-change** property.
ldap-server-definition-uri	String/URI	Required if **authentication-type** is **"ldap"**	The value to be set as the user's **ldap-server-definition-uri** property.
userid-on-ldap-server	String	Optional	The value to be set as the user's **userid-on-ldap-server** property.
session-timeout	Integer	Optional	The value to be set as the user's **session-timeout** property.
verify-timeout	Integer	Optional	The value to be set as the user's **verify-timeout** property.
idle-timeout	Integer	Optional	The value to be set as the user's **idle-timeout** property.
min-pw-change-time	Integer	Optional	The value to be set as the user's **min-pw-change-time** property.
max-failed-logins	Integer	Optional	The value to be set as the user's **max-failed-logins** property.
disable-delay	Integer	Optional	The value to be set as the user's **disable-delay** property.
inactivity-timeout	Integer	Optional	The value to be set as the user's **inactivity-timeout** property.
disruptive-pw-required	Boolean	Optional	The value to be set as the users **disruptive-pw-required** property.
disruptive-text-required	Boolean	Optional	The value to be set as the user's **disruptive-text-required** property.
allow-remote-access	Boolean	Optional	The value to be set as the user's **allow-remote-access** property.
allow-management-interfaces	Boolean	Optional	The value to be set as the user's **allow-management-interfaces** property.
max-web-services-api-sessions	Integer	Optional	The value to be set as the user's **max-web-services-api-sessions** property.
web-services-api-session-idle-timeout	Integer	Optional	The value to be set as the user's **web-services-api-session-idle-timeout** property.
multi-factor-authentication-required	Boolean	Optional	The value to be set as the user's **multi-factor-authentication-required** property.

Field name	Type	Rqd/Opt	Description
`mfa-types`	Array of String Enum	Optional	The value to be set as the user's **mfa-types** property.
`primary-mfa-server-definition-uri`	String/URI	Required if **mfa-types** contains **"mfa-server"**	The value to be set as the user's **primary-mfa-server-definition-uri** property.
`backup-mfa-server-definition-uri`	String/URI	Optional	The value to be set as the user's **backup-mfa-server-definition-uri** property.
`mfa-policy`	String	Required if **mfa-types** contains **"mfa-server"**	The value to be set as the user's **mfa-policy** property.
`mfa-userid`	String	Optional	The value to be set as the user's **mfa-userid** property.
`mfa-userid-override`	String	Optional	The value to be set as the user's **mfa-userid-override** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/URI	Canonical URI path of the new User object.

Description

This operation creates a new console user.

On successful execution of this operation the user is created using the inputs as specified by the request body. The URI of the new user is provided in the response body and in a **Location** response header as well. An Inventory Change notification is emitted asynchronously.

The request body is validated against the schema described in the "Request body contents" on page 911. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **password-rule-uri** when the **authentication-type** value is **"ldap"**) or the specified name is not unique. If a URI in the request body does not designate an existing resource of the appropriate type, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Users** task to create a standard user or the **Manage User Templates** task to create a template user; otherwise, status code 403 (Forbidden) is returned.

Certain user names are used internally by the Hardware Management Console and are therefore not available for use when creating a new user. An attempt to create a user with one of these names results in status code 400 (Bad Request) indicating that there is already a user with that name. The list of such names is case-insensitive and includes the following:

- SOOACSADMIN
- SOOADVANCED
- SOOENSADMIN
- SOOENSOPERATOR

- SOOOPERATOR
- SOOSERVICE
- SOOSTORAGEADMIN
- SOOSYSPROG
- PEDEBUG

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to create a standard user or the **Manage User Templates** task to create a template user.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 913, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 462. Create User: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A user with the name specified in the request body already exists.
	311	The password does not conform to the requirements of the password policy in effect for this user.
	337	The **primary-mfa-server-definition-uri** in the request body and the **backup-mfa-server-definition-uri** in the request body designate the same MFA Server Definition object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	323	The **password-rule-uri** field in the request body does not designate an existing Password Rule object.
	324	The **ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	338	The **primary-mfa-server-definition-uri** in the request body does not designate an existing MFA Server Definition object.
	339	The **backup-mfa-server-definiton-uri** in the request body does not designate an existing MFA Server Definition object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/users HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 234
{
   "authentication-type":"local",
   "description":"Gabby McRosie - company president",
   "name":"Gabby",
   "password":"abc123pw",
   "password-rule-uri":"/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
   "type":"standard"
}
```

Figure 482. Create User: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:29 GMT
content-type: application/json;charset=UTF-8
content-length: 64
{
   "object-uri":"/api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91"
}
```

Figure 483. Create User: Response

Delete User

The `Delete User` operation deletes a User object designated by its object ID. This operation is not valid for pattern-based users.

HTTP method and URI

`DELETE /api/users/`*`{user-id}`*

In this request, the URI variable *{user-id}* is the object ID of the User object to be deleted.

Description

This operation removes a specified user from the console. The user is identified by the *{user-id}* variable in the URI.

Upon successfully removing the user, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously.

The URI path must designate an existing User object; otherwise, status code 404 (Not Found) is returned. If the API user does not have user-related-access permission to the designated User object or action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 404 (Not Found) is returned. If the API user has user-related-access permission to the designated User object but not action/task permission to the **Manage Users** or **Manage User Templates** task, whichever is appropriate, status code 403 (Forbidden) is returned. It is an error for an API user to attempt to delete his own User object; any attempt to do so results in status code 400 (Bad Request). If the request URI identifies a template user, and a user or a User Pattern refers to that template user, the request fails and status code 409 (Conflict) is returned. An attempt to delete a pattern-based user is not valid and fails with status code 400 (Bad Request).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Users** task to delete a non-template user, or the **Manage User Templates** task to delete a template user.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 463. Delete User: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	312	This operation is not supported for an object of this type. Pattern-based users may not be deleted.
	313	The request URI designates the API user's own User object.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users or User Patterns refer to this template user.
	320	The object cannot be deleted at this time. It is currently identified as the Automatic Logon ID for the Hardware Management Console.
	321	The object cannot be deleted at this time. It is currently the only locally-authenticated user with permission to the tasks for managing users and user roles.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 484. Delete User: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT

<No response body>
```

Figure 485. Delete User: Response

Inventory service data

Information about the users managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"user"** are to be included. An entry for a particular user is included only if the API user has access permission to that object as described in the `Get User Properties` operation.

For each User object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get User Properties` operation. That is, the data provided is the same as would be provided if a `Get User Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single user. This object would appear as one array entry in the response array:

```
{
   "allow-management-interfaces": false,
   "allow-remote-access": false,
   "authentication-type": "ldap",
   "class": "user",
   "default-group-uri": "/api/groups/fc400fc6-54e3-335d-854f-a0a6d10f000b",
   "description": "System admnistrator",
   "disable-delay": 1,
   "disabled": false,
   "disruptive-pw-required": true,
   "disruptive-text-required": false,
   "email-address":"finn@example.com",
   "force-password-change": null,
   "force-shared-secret-key-change":false,
   "idle-timeout": 0,
   "inactivity-timeout": 0,
   "is-locked": false,
   "ldap-server-definition-uri": "/api/console/ldap-server-definitions/
      4927787e-34c4-11e4-a1ea-5ef3fcae8020",
   "max-failed-logins": 3,
   "max-web-services-api-sessions": 100,
   "min-pw-change-time": null,
   "multi-factor-authentication-required":true,
   "name": "sysadmin",
   "object-id": "9069f7a6-34c5-11e4-af4d-5ef3fcae8020",
   "object-uri": "/api/users/9069f7a6-34c5-11e4-af4d-5ef3fcae8020",
   "parent": "/api/console",
   "password-expires": -1,
   "password-rule-uri": null,
   "replication-overwrite-possible": false,
   "session-timeout": 0,
   "type": "standard",
   "user-roles": [
      "/api/user-roles/b39afb87-d915-4070-a22f-91b158c6c01e"
   ],
   "userid-on-ldap-server": "sysadmin",
   "verify-timeout": 15,
   "web-services-api-session-idle-timeout": 15
}
```


Figure 486. User object: Sample inventory data

User Role object

A User Role object represents an authority role which can be assigned to one or more console users. A role may allow access to specific managed objects, classes of managed objects, groups and/or tasks. There are two types of User Roles: user-defined and system-defined. User-defined User Roles are created by a console user, whereas the system-defined User Roles are pre-defined, standard User Roles supplied with the console.

Through user-related-access permission described in "User-related-access permission" on page 892, API users are permitted to see certain User Role objects in a `List User Roles` response and issue `Get User Role Properties` for those User Role objects. An API user with action/task permission to the `Manage User Roles` task is permitted to view any User Role object and change any user-defined User Role object.

System-defined user roles

There are many system-defined User Roles supplied with the console. System-defined roles may not be modified or deleted. The names assigned to the system-defined roles are enumerated in Appendix C, "Enum values for the User Role object," on page 1439.

Data model

This object includes the properties defined in the "Base managed object properties schema" on page 100, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

The following class-specific specializations apply to the other base managed object properties:

Table 464. User Role object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the User Role object is of the form /api/user-roles/*{user-role-id}*, where *{user-role-id}* is the value of the **object-id** property of the User Role object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a User Role object is **"user-role"**.
name	(ro)	String	The name of the User Role object. This name must be unique among all of the console's User Roles with the same **type** value. While pre-existing User Role names are virtually unrestricted in terms of length and characters, new User Role names must conform to the length and character requirements of the **name** property described in the "Base managed object properties schema" on page 100. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new User Role object. The names of system-defined User Roles are listed in Appendix C, "Enum values for the User Role object," on page 1439.
description	(w)(pc)	String (0-1024)	The description of the User Role object. Default: an empty string

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 465. User Role object: class specific additional properties

Name	Qualifier	Type	Description
type	—	String Enum	The type of User Role. Supported values are: • **"user-defined"** • **"system-defined"**
associated-system-defined-user-role-uri	(w)(pc)	String/ URI	Canonical URI path of the system-defined User Role associated with this User Role. The User Roles which are valid associated User Roles are identified in Appendix C, "Enum values for the User Role object," on page 1439. If this is a system-defined User Role, this property is null, because system-defined User Roles do not have an associated User Role. Default: the URI path of the Operator Tasks system-defined User Role.

Table 465. User Role object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
permissions	(c)(pc)	Array of objects	The list of permissions included in this User Role. These permissions are identified by permission-info objects, as defined in the next table. The members provided in this list can change as a result of the `Add Permission to User Role` and `Remove Permission from User Role` operations. If there are no permissions in this User Role, an empty array is provided. Default: an empty array
is-inheritance-enabled	(w)(pc)	Boolean	Indicates whether User Role inheritance is enabled. When **true**, if this User Role permits access to a parent managed object, then all managed objects that are hosted by the parent managed object are also permitted by this role. When **false**, no such inherited permissions exist. Default: **false**.
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

Each nested permission-info object contains the following fields:

Table 466. permission-info object properties		
Name	**Type**	**Description**
permitted-object	String/ URI or String Enum	Canonical URI path or String Enum which identifies the object(s) or task to which permission has been granted. Granting permission to a Task object gives the user action/task permission to the console task associated with that Task object, and the user is permitted to target any of their authorized objects with that task. If this identifies a class of managed objects, a specific managed object or a Group object, then permission is granted to view the details of the object(s) and target the object(s) with any authorized task for which it is an appropriate target. This field is one of the following: • The identifier for a class of managed objects. This identifier is the String Enum value for the class from Appendix B, “Enum values for a type of managed objects within User Roles,” on page 1437. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI "/api/system-manual-definition", which denotes a specific managed object known on the HMC UI as "System Manual Definition"[1] The type of object(s) is indicated by the **permitted-object-type** property.

Table 466. permission-info object properties (continued)		
Name	**Type**	**Description**
permitted-object-type	String Enum	Identifies the type of object(s) identified by the **permitted-object** property. • **"object"** - **permitted-object** contains a URI that identifies one of the following: – A specific managed object – A Group object – A Task object. • **"object-class"** - **permitted-object** contains a String Enum value fromAppendix B, "Enum values for a type of managed objects within User Roles," on page 1437.
include-members	Boolean	Indicates whether the members of a group are included in the User Role, or if only the group itself is included. True if members are included; false otherwise.
view-only-mode	Boolean	Indicates whether it is a task's view-only version that is in this User Role. Only certain tasks support a view-only mode. This field is only provided if the **permitted-object** field identifies such a Task object. A User Role cannot have both the view-only version and the non-view-only version in its set of permissions. Prerequisite: the **permitted-object** field identifies a Task object that supports a view-only mode.
[1]The object identified by this special URI does not have full API support. It is only valid as a permitted object identifier in a User Role object. Thus it may be included in the `Get User Role Properties` response body, and it is valid in the request body of the `Add Permission to User Role` and `Remove Permission from User Role` operations.		

List User Roles

The `List User Roles` operation lists User Roles defined to the console.

HTTP method and URI

```
GET /api/console/user-roles
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid **type** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`user-roles`	Array of objects	Array of nested user-role-info objects as described in the next table.

Each nested user-role-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The **object-uri** property of the User Role object.
`name`	String	The **name** property of the User Role object.
`type`	String Enum	The **type** property of the User Role object.

Description

The `List User Roles` operation lists User Roles defined to the console. Some basic properties are provided for each user role.

If the **name** query parameter is specified, the returned list is limited to those User Roles that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid User Role **type** property value. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those User Roles that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A User Role is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage User Roles** task. If there is a User Role to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no User Roles defined to the console or if no User Roles are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Role objects included in the response body or action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 921.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage notes

While it is intended that system-defined User Roles have a name that begins with "hmc-", there is no such guarantee. API clients are cautioned to use the **type** property rather than **name** to reliably distinguish between system-defined and user-defined User Roles.

Example HTTP interaction

```
GET /api/console/user-roles?name=.*admin.*&type=system-defined HTTP/1.1
x-api-session: 40hwdiopt6avosjzkk4v20sac85xtd842klcjh9hox3wfh8f89
```

Figure 487. List User Roles: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 21:12:08 GMT
content-type: application/json;charset=UTF-8
content-length: 556
{
   "user-roles":[
      {
         "name":"hmc-access-administrator-tasks",
         "object-uri":"/api/user-roles/11182f6f-f24a-4aea-b18b-a2b2be46bdf1",
         "type":"system-defined"
      },
      {
         "name":"hmc-energy-administrator-tasks",
         "object-uri":"/api/user-roles/c4888502-72c7-4507-9c1b-9df9a24128ae",
         "type":"system-defined"
      },
      {
         "name":"hmc-storage-administrator-objects",
         "object-uri":"/api/user-roles/6ef9ab39-4195-4035-a5e0-00e91944352a",
         "type":"system-defined"
      },
      {
         "name":"hmc-storage-administrator-tasks",
         "object-uri":"/api/user-roles/9f197c15-6eb4-4b1c-94d9-b38ffa5861ab",
         "type":"system-defined"
      }
   ]
}
```

Figure 488. List User Roles: Response

Get User Role Properties

The `Get User Role Properties` operation retrieves the properties of a single User Role object that is designated by its object ID.

HTTP method and URI

```
GET /api/user-roles/{user-role-id}
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role object whose properties are to be retrieved.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User Role object as defined in the data model section. Field names and data types

in the JSON object are the same as the property names and data types defined in the "Data model" on page 918.

Description

This operation returns the current properties of a single User Role object that is designated by *{user-role-id}*.

On successful execution, all of the current properties as defined in the data model for the User Role object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User Role object and the API user must have user-related-access permission to it or action/task permission to the **Manage User Roles** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Role object specified in the request URI, or action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 923.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage notes

While it is intended that system-defined User Roles have a name that begins with "hmc-", there is no such guarantee. API clients are cautioned to use the **type** property rather than **name** to reliably distinguish between system-defined and user-defined User Roles.

Example HTTP interaction

```
GET /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 489. Get User Role Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT
content-type: application/json;charset=UTF-8
content-length: 847
{
   "associated-se-user-role-uri":null,
   "class":"user-role",
   "description":"Role for managing department business",
   "is-inheritance-enabled":false,
   "is-locked":false,
   "name":"Dept Admin",
   "object-id":"eb53f840-4a7a-11e4-affa-1c6f65065a91",
   "object-uri":"/api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91",
   "parent":"/api/console",
   "permissions":[
      {
         "permitted-object":"/api/console/tasks/4d5a39f0-c1df-4a2e-9a46-5bf6f6a759f3",
         "permitted-object-type":"object",
         "view-only-mode":false
      },
      {
         "include-members":true,
         "permitted-object":"/api/groups/cafb3a9b-6a34-4475-938f-98c5d60868a5",
         "permitted-object-type":"object"
      },
      {
         "permitted-object":"lpar-image",
         "permitted-object-type":"object-class"
      },
      {
         "permitted-object":"/api/console/tasks/45042443-7202-40b3-8630-b7a563a21d8d",
         "permitted-object-type":"object"
      }
   ],
   "replication-overwrite-possible":false,
   "type":"user-defined"
}
```

Figure 490. Get User Role Properties: Response

Update User Role Properties

The `Update User Role Properties` operation updates the properties of a single user-defined User Role object that is designated by its object ID. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/user-roles/{user-role-id}
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation

Description

This operation updates writable properties of the User Role object specified by *{user-role-id}* .

The URI path must designate an existing User Role object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. An attempt to update a system-defined User Role is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation.

If the update changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 467. Update User Role Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be updated.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 99
{
   "description":"A new and improved description of this User Role",
   "is-inheritance-enabled":true
}
```

Figure 491. Update User Role Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 492. Update User Role Properties: Response

Add Permission to User Role

The `Add Permission to User Role` operation adds a specified permission to a specified user-defined User Role thereby granting that permission to all users that have that User Role. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/user-roles/{user-role-id}/operations/add-permission
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role to which a permission is to be added.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`permitted-object`	String/ URI or String Enum	Required	Canonical URI path or String Enum which identifies the object(s) or task to which permission is to be granted. See the data model for more information about permissions. This field is one of the following: • The identifier for a class of managed objects. This identifier is the String Enum value for the class from Appendix B, "Enum values for a type of managed objects within User Roles," on page 1437. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI "/api/system-manual-definition", which denotes a specific managed object known on the HMC UI as "System Manual Definition" The type of object(s) is indicated by the **permitted-object-type** field.
permitted-object-type	String Enum	Required	Identifies the type of object(s) identified by the **permitted-object** field. Supported values are: • **"object"** - **permitted-object** contains a URI that identifies one of the following: – A specific managed object – A Group object – A Task object. • **"object-class"** - **permitted-object** contains a String Enum value from Appendix B, "Enum values for a type of managed objects within User Roles," on page 1437.
include-members	Boolean	Optional	Indicates whether the members of the group are included in the User Role, or if only the group itself is included. True if members are included; false otherwise. Default: false
view-only-mode	Boolean	Optional	Indicates whether it is the task's view-only version that is being added to this User Role. Only certain tasks support a view-only mode. This field is only allowed if the **permitted-object** field identifies such a Task object. Prerequisite: the **permitted-object** field identifies a Task object that supports a view-only mode. Default: true

Description

This operation adds a permission to a User Role

On successful execution of this operation the permission specified in the request body has been added to the User Role identified in the request URI.

The request body is validated against the schema described in the "Request body contents" on page 927. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User Role object,

status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. If the URI in the request body does not designate an existing resource, status code 404 (Not Found) is returned. An attempt to alter a system-defined User Role is not valid and fails with status code 400 (Bad Request).

If the specified permission is already in the User Role, or an attempt is made to have both the view-only and non-view-only versions of a Task in the User Role, status code 409 (Conflict) is returned.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 468. Add Permission to User Role: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be altered.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	315	The permission identified in the request body is already in the User Role.
	327	The view-only and non-view-only versions of a Task cannot be in the same User Role. An attempt was made to add one version when the other was already in the User Role's set of permitted objects.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91/operations/
   add-permission HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 114
{
   "permitted-object":"/api/console/tasks/5d8c9f60-a2b3-4327-9fd4-791df8a60dcc",
   "permitted-object-type":"object"
}
```

Figure 493. Add Permission to User Role: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 494. Add Permission to User Role: Response

Remove Permission from User Role

The `Remove Permission from User Role` operation removes a specified permission from a specified user-defined User Role. System-defined User Roles are immutable; therefore, this operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/user-roles/{user-role-id}/operations/remove-permission
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role from which a permission is to be removed.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`permitted-object`	String/ URI or String Enum	Required	Canonical URI path or String Enum which identifies the object(s) or task to which permission is to be removed. See the data model for more information about permissions. This field is one of the following: • The identifier for a class of managed object. This identifier is the String Enum value for the class from Appendix B, "Enum values for a type of managed objects within User Roles," on page 1437. • The URI of a specific managed object. • The URI of a Group object. • The URI of a Task object. • The well-known URI "/api/system-manual-definition", which denotes a specific managed object known on the HMC UI as "System Manual Definition" The type of object(s) is indicated by the **permitted-object-type** field.
`permitted-object-type`	String Enum	Required	Identifies the type of object(s) identified by the **permitted-object** field. Supported values are: • **"object" - permitted-object** contains a URI that identifies one of the following: – A specific managed object. – A Group object. – A Task object. • **"object-class" - permitted-object** contains a String Enum value from Appendix B, "Enum values for a type of managed objects within User Roles," on page 1437.
`include-members`	Boolean	Optional	Indicates whether the members of the group are included in this operation, or if only the group itself is included. True if members are included; false otherwise. Prerequisite: the **permitted-object** field identifies a Group object. Default: false
`view-only-mode`	Boolean	Optional	Indicates whether it is permission to a task's view-only version that is to be removed from this User Role. Only certain tasks support a view-only mode. This field is only allowed if the **permitted-object** field identifies such a Task object. Prerequisite: the **permitted-object** field identifies a Task object that supports a view-only mode. Default: true

Description

This operation removes a permission from a User Role

On successful execution of this operation the permission specified in the request body has been removed from the User Role identified in the request URI.

The request body is validated against the schema described in "Request body contents" on page 930. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the request URI does not designate an existing User Role object, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. If the URI in the request body does not designate an existing resource, status code 404 (Not Found) is returned. An attempt to alter a system-defined User Role is not valid and fails with status code 400 (Bad Request).

If the specified permission is not in the User Role, or if removing it would leave one or more users without object-access permission to their default group, status code 409 (Conflict) is returned.

If this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 469. Remove Permission from User Role: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined User Roles may not be altered.
	316	The permission identified in the request body is not in the User Role.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
	2	A URI in the request body does not designate an existing resource of the correct type.
409 (Conflict)	316	The permission identified in the request body is not in the User Role.
	328	The permission cannot be removed at this time, because doing so would leave one or more users without object-access permission to their default group.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91/operations/
   remove-permission HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 114
{
   "permitted-object":"/api/console/tasks/5d8c9f60-a2b3-4327-9fd4-791df8a60dcc",
   "permitted-object-type":"object"
}
```

Figure 495. Remove Permission from User Role: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 496. Remove Permission from User Role: Response

Create User Role

The `Create User Role` operation creates a user-defined User Role object with the given properties on the console. This operation is not valid for system-defined User Roles.

HTTP method and URI

```
POST /api/console/user-roles
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the User Role's **name** property.
`description`	String	Optional	The value to be set as the User Role's **description** property.
`associated-system-defined-user-role-uri`	String/URI	Optional	The value to be set as the User Role's **associated-system-defined-user-role-uri** property.
`is-inheritance-enabled`	Boolean	Optional	The value to be set as the User Role's **is-inheritance-enabled** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path of the new User Role object.

Description

This operation creates a new, empty, user-defined User Role.

On successful execution of this operation the User Role is created using the inputs as specified by the request body. The URI of the new User Role is provided in the response body and in a **Location** response header as well. An Inventory Change notification is emitted asynchronously. The `Add Permission to User Role` operation can then be used to add permissions to the new User Role.

The request body is validated against the schema described in "Request body contents" on page 933. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the specified **name** is not unique, status code 400 (Bad Request) is returned. In addition, the API user must have action/task permission to the **Manage User Roles** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 933, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 470. Create User Role: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A User Role with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	2	A URI in the request body does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59 and the Location response header contains the URI of the newly created object.

Example HTTP interaction

```
POST /api/console/user-roles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 78
{
   "description":"Role for managing department business",
   "name":"Dept Admin"
}
```

Figure 497. Create User Role: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:31 GMT
content-type: application/json;charset=UTF-8
content-length: 69
{
   "object-uri":"/api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91"
}
```

Figure 498. Create User Role: Response

Delete User Role

The `Delete User Role` operation deletes a user-defined User Role object designated by its object ID. This operation is not valid for system-defined User Roles.

HTTP method and URI

```
DELETE /api/user-roles/{user-role-id}
```

In this request, the URI variable *{user-role-id}* is the object ID of the User Role object to be deleted.

Description

This operation removes a specified User Role from the console. The User Role is identified by the *{user-role-id}* variable in the URI.

Upon successfully removing the User Role, HTTP status code 204 (No Content) is returned and no response body is provided. An Inventory Change notification is emitted asynchronously.

The URI path must designate an existing User Role object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Role object or action/task permission to the **Manage User Roles** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Role object but not action/task permission to the **Manage User Roles** task, status code 403 (Forbidden) is returned. An attempt to delete a system-defined User Role is not valid and fails with status code 400 (Bad Request). If any user has the specified User Role, the request fails and status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Roles** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 471. Delete User Role: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	312	This operation is not supported for an object of this type. System-defined User Roles may not be deleted.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this User Role.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/user-roles/eb53f840-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 499. Delete User Role: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:32 GMT

<No response body>
```

Figure 500. Delete User Role: Response

Inventory service data

Information about the User Roles managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User Role objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"user-role"** are to be included. An entry for a particular User Role is included only if the API user has access permission to that object as described in the `Get User Role Properties` operation.

For each User Role object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get User`

`Role Properties` operation. That is, the data provided is the same as would be provided if a `Get User Role Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single User Role. This object would appear as one array entry in the response array:


```
{
    "associated-system-defined-user-role-uri": "/api/user-roles/b39afb87-d915-
       4070-a22f-91b158c6c01e",
    "class": "user-role",
    "description": "Role that allows management of users",
    "is-inheritance-enabled": false,
    "is-locked": false,
    "name": "user_roles_9",
    "object-id": "db7f9448-3737-11e4-a5fc-5ef3fcae8020",
    "object-uri": "/api/user-roles/db7f9448-3737-11e4-a5fc-5ef3fcae8020",
    "parent": "/api/console",
    "permissions": [
       {
          "permitted-object": "/api/console/tasks/f8d653f4-eab2-4547-97c0-
             a26f762218ba",
          "permitted-object-type": "object"
       },
       {
          "permitted-object": "/api/console/tasks/36e32fb4-7b60-4677-b462-
             e786f337ea0f",
          "permitted-object-type": "object"
       },
       {
          "permitted-object": "/api/console/tasks/8ef2b7ca-c2d2-4a5b-9d52-
             b4d1a28ccb15",
          "permitted-object-type": "object"
       }
    ],
    "replication-overwrite-possible": false,
    "type": "user-defined"
}
```


Figure 501. User Role object: Sample inventory data

Task object

A Task object is an element of the console object and represents an action that a console user with appropriate authority can perform. These actions could be available through the console's graphical user interface, the Web Services APIs or both. Tasks are predefined by the console and cannot be created, modified or deleted.

All API users have object-access permission to all Tasks and thus are permitted to issue `List Tasks` and `Get Task Properties` for any Task object. Note that this object-access permission to a Task object does not give an API user action/task permission to the task represented by the Task object.

Data model

The Task object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 98.

Table 472. Task object: properties			
Name	**Qualifier**	**Type**	**Description**
element-uri	—	String/ URI	The canonical URI path of the Task object is of the form /api/console/tasks/*{task-id}*, where *{task-id}* is the value of the **element-id** property of the Task object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a Task object is **"task"**.
name	(ro)	String Enum	The name of the Task object. The task names are documented in Appendix D, "Enum values for the Task object," on page 1441.
description	—	String (0-1024)	The description of the Task object. Default: an empty string
view-only-mode-supported	—	Boolean	Indicates whether this task supports a view-only mode.

List Tasks

The `List Tasks` operation lists Tasks defined to the console.

HTTP method and URI

```
GET /api/console/tasks
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`tasks`	Array of objects	Array of nested task-info objects as described in the next table.

Each nested task-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the Task object.
`name`	String	The **name** property of the Task object.

Description

This operation lists tasks defined to the console. Some basic properties are provided for each task.

If the **name** query parameter is specified the returned list is limited to those tasks that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If no tasks are to be included in the results due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 938.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/tasks?name=manage-user."*" HTTP/1.1
x-api-session: 3hrke2gqugoxejhxenpcdd022kff8iwx10opa33yiv3vse17pd
```

Figure 502. List Tasks: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 09 Sep 2019 18:27:18 GMT
content-type: application/json;charset=UTF-8
content-length: 417
{
   "tasks":[
      {
         "element-uri":"/api/console/tasks/4d3beaea-8971-4ea2-a96d-caff8e60cb10",
         "name":"manage-user-roles"
      },
      {
         "element-uri":"/api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7",
         "name":"manage-users"
      },
      {
         "element-uri":"/api/console/tasks/89c2501e-469a-4b95-81d7-188198e8528f",
         "name":"manage-user-templates"
      },
      {
         "element-uri":"/api/console/tasks/e5b286dd-8089-448b-a01e-d6261e84a283",
         "name":"manage-user-patterns"
      }
   ]
}
```

Figure 503. List Tasks: Response

Get Task Properties

The `Get Task Properties` operation retrieves the properties of a single Task object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/tasks/{task-id}
```

In this request, the URI variable *{task-id}* is the element ID of the Task object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the Task object as defined in the data model section. Field names and data types in the JSON object are the same as the property names and data types defined in the "Data model" on page 937.

Description

This operation returns the current properties of a single Task object specified by *{task-id}* .

On successful execution, all of the current properties as defined in the data model for the Task object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Task object; otherwise, status code 404 (Not Found) is returned.

Authorization requirements

This operation has no explicit authorization requirements.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 940.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7 HTTP/1.1
x-api-session: 69prd5pgk20edka08dmxjaqp0iyqpu0ixiq2028tduivi242yh
```

Figure 504. Get Task Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 09 Sep 2019 18:28:56 GMT
content-type: application/json;charset=UTF-8
content-length: 268
{
   "class":"task",
   "description":"Create, modify, and delete users",
   "element-id":"8a522e34-ca08-475a-9d71-90100f2316a7",
   "element-uri":"/api/console/tasks/8a522e34-ca08-475a-9d71-90100f2316a7",
   "name":"manage-users",
   "parent":"/api/console",
   "view-only-mode-supported":false
}
```

Figure 505. Get Task Properties: Response

Inventory service data

Information about the tasks managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Task objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included.

For each Task object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get Task Properties` operation. That is, the data provided is the same as would be provided if a `Get Task Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single Task. This object would appear as one array entry in the response array:

```
{
   "class": "task",
   "description": "Customize the Application Programming Interface for the
      console",
   "element-id": "ee76dca1-9aee-4530-bf66-667ae728ed10",
   "element-uri": "/api/console/tasks/ee76dca1-9aee-4530-bf66-667ae728ed10",
   "name": "customize-api-settings",
   "parent": "/api/console",
   "view-only-mode-supported": false
}
```

Figure 506. Task object: Sample inventory data

User Pattern object

User Patterns and user templates allow a system administrator to define a group of console users at once whose user IDs all match a certain pattern (for example, a regular expression) and who have a certain set of attributes. A User Pattern object is an element of the console object and defines a pattern for user IDs that are not defined to the console but can be verified by an LDAP server for user authentication. Each pattern identifies a template User object which defines many characteristics of such users. A successful logon with a user ID that matches a User Pattern results in the creation of a pattern-based user, with many of its attributes coming from the associated template. User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object operation "Reorder User Patterns" on page 822.

Through user-related-access permission described in "User-related-access permission" on page 892, API users are permitted to see certain User Pattern objects in a `List User Patterns` response and issue `Get User Pattern Properties` for those User Pattern objects. An API user with action/task permission to the `Manage User Patterns` task is permitted to view and change any User Pattern object and change the pattern search order.

Data model

This object contains the following properties. Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **template-name-override** is **null** when the **ldap-server-definition-uri** value is **null**.

There are 3 mutually exclusive methods to identify the user template to be used when a user pattern is used during logon. Exactly one of the following methods must be used when the User Pattern is created:

- Method 1: Specify a specific template (**specific-template-uri** property)
- Method 2: Specify a template name override attribute; an optional default template can also be specified. (**template-name-override-ldap-server-definition-uri**, **template-name-override-default-template-uri** and **template-name-override** properties)
- Method 3: Specify that LDAP group membership is to be used to identify the template; an optional default template can also be specified. (**ldap-group-to-template-mappings**, **ldap-group-ldap-server-definition-uri**, **ldap-group-default-template-uri** properties)

The preferred properties to use are: **specific-template-uri**, **template-name-override-ldap-server-definition-uri**, **template-name-override-default-template-uri**, **ldap-group-to-template-mappings**, **ldap-group-ldap-server-definition-uri**, **ldap-group-default-template-uri**, and **domain-name-restrictions-ldap-server-definition-uri**; they are preferred over **user-template-uri** and **ldap-server-definition-uri**. Either those preferred properties or the others, but not both, may be used on a `Create User Pattern` and `Update User Pattern Properties` operation. As changes are made to those properties, the APIs will adjust properties to keep them in a consistent state. If a User Pattern

is configured to use method 2 or 3 and no default template is specified, then the **user-template-uri** property will be **null**.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

Table 473. User Pattern object: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the User Pattern object is of the form /api/console/user-patterns/*{user-pattern-id}*, where *{user-pattern-id}* is the value of the **element-id** property of the User Pattern object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a User Pattern object is **"user-pattern"**.
name	(w)	String	The name of the User Pattern object. This name must be unique among all User Patterns on the console. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new User Pattern object.
description	(w)	String (0-1024)	The description of the User Pattern object. Default: an empty string
pattern	(w)	String	The actual User Pattern expression. The combination of **pattern** and **type** must be unique on the console. Must not be an empty string.
type	(w)	String Enum	The style in which this pattern is expressed, which is one of the following values: • **"glob-like"** - The pattern may include limited special characters: an asterisk in the pattern matches zero or more characters in the user ID, and a question mark in the pattern matches any single character in the user ID. • **"regular-expression"** - The pattern is expressed as a UNIX regular expression.
search-order-index	—	Integer	A zero-based index position of this User Pattern in the pattern search order used during logon processing.
retention-time	(w)	Integer (0-2147483647)	The time in days that the user data for pattern-based users created based on this pattern will be retained. A value of 0 indicates not to retain the settings.
user-template-uri[1]	(w)	String/ URI	The canonical URI path of the User object containing the template definition that applies when a successful match occurs using this User Pattern. Default: **null**

Table 473. User Pattern object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ldap-server-definition-uri[1,2]	(w)	String/URI	The canonical URI path of the LDAP Server Definition object which identifies the LDAP server to be used to fetch the template name and/or domain names mentioned in the **template-name-override** and **domain-name-restrictions** properties when processing a logon for a user ID that matches this pattern, or null if the LDAP server in the template is used.
template-name-override	(w)	String	The name of the LDAP attribute that contains the name of the user template definition for the user, or **null** if there is no such attribute. When not **null**, this property is used in an LDAP lookup to override the user template identified in the **user-template-uri** field, and it must not be an empty string. Prerequisite: **ldap-server-definition-uri** or **template-name-override-ldap-server-definition-uri** is not **null** Default: **null**
domain-name-restrictions	(w)	String	The name of the LDAP attribute that contains the information about which consoles the user is allowed to log onto, or **null** if there is no such attribute. Must not be an empty string. Prerequisite: **ldap-server-definition-uri** or **domain-name-restrictions-ldap-server-definition-uri** is not **null** Default: **null**
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.
specific-template-uri[1]	(w)	String/URI	The canonical URI path of the User object containing the template definition that applies when a successful match occurs using this User Pattern and no LDAP group lookup or template name override lookup is specified. Default: **null**
template-name-override-ldap-server-definition-uri[1,2]	(w)	String/URI	The canonical URI path of the LDAP Server Definition object which identifies the LDAP server to be used to fetch the attribute, if any, identified in the **template-name-override** property when processing a logon for a user ID that matches this pattern. Must be **null** if **template-name-override** is **null**; otherwise, it must not be **null**. Default: **null**
template-name-override-default-template-uri[1]	(w)	String/URI	The canonical URI path of the User object containing the template definition that applies when a successful match occurs using this User Pattern and a template name override lookup is specified but did not yield a template name. If **null** in this case, the logon will fail. Default: **null**

Table 473. User Pattern object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ldap-group-to-template-mappings[1]	(w)	Array of group-to-template-mapping objects	An ordered list mapping LDAP groups to user templates, or **null** if LDAP group membership checks are not to be performed during logon processing. Each entry of the list maps an LDAP group name to a User object URI, where the User object is of **type "template"**. Must be **null** if **ldap-group-ldap-server-definition-uri** is **null**; otherwise, it must not be **null** or an empty array. Default: **null**
ldap-group-ldap-server-definition-uri[1]	(w)	String/ URI	The canonical URI path of the LDAP Server Definition object for the LDAP server used for group membership checks during logon processing, or **null** if LDAP group membership checks are not to be performed. Must be **null** if **ldap-group-to-template-mappings** is **null**; otherwise, it must not be **null**. Default: **null**
ldap-group-default-template-uri[1]	(w)	String/ URI	The canonical URI path of the User object containing the template definition that applies when a successful match occurs using this User Pattern and an LDAP group lookup is specified but did not yield a template name. If **null** in this case, the logon will fail. Default: **null**
domain-name-restrictions-ldap-server-definition-uri[1,2]	(w)	String/ URI	The canonical URI path of the LDAP Server Definition object which identifies the LDAP server to be used to fetch the attribute, if any, identified in the **domain-name-restrictions** property when processing a logon for a user ID that matches this pattern. Must be **null** if **domain-name-restrictions** is **null**; otherwise, it must not be **null**. Default: **null**

[1]The preferred properties to use to identify the user template are: **specific-template-uri**, **template-name-override-ldap-server-definition-uri**, **template-name-override-default-template-uri**, **ldap-group-to-template-mappings**, **ldap-group-ldap-server-definition-uri**, **ldap-group-default-template-uri**, and **domain-name-restrictions-ldap-server-definition-uri**; they are preferred over **user-template-uri** and **ldap-server-definition-uri**. Either those preferred properties or the others, but not both, may be used on a `Create User Pattern` and `Update User Pattern Properties` operation. As changes are made to those properties, the APIs will adjust properties to keep them in a consistent state. If a User Pattern is configured to use a template name override lookup or a group membership lookup and no default template is specified, then the **user-template-uri** property will be **null**.

[2]This User Pattern property must not refer to an LDAP Server Definition with an **authentication-type** of **"direct"**, because such LDAP Server Definitions are intended for performing group membership checks.

Table 474. group-to-template-mapping properties		
Name	**Type**	**Description**
ldap-group-name	String	The name of the LDAP group
template-uri	String/ URI	The canonical URI of the user template associated with the LDAP group.

List User Patterns

The `List User Patterns` operation lists User Patterns defined to the console.

HTTP method and URI

```
GET /api/console/user-patterns
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid **type** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`user-patterns`	Array of objects	Array of nested user-pattern-info objects as described in the next table.

Each nested user-pattern-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the User Pattern object.
`name`	String	The **name** property of the User Pattern object.
`type`	String Enum	The **type** property of the User Pattern object.

Description

This operation lists User Patterns defined to the console. Some basic properties are provided for each User Pattern.

A User Pattern is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage User Patterns** task. If there is a User Pattern to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no User Patterns defined to the console or if no User Patterns are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Pattern objects included in the response body or action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 946.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage notes

User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object operation "Reorder User Patterns" on page 822. The `List User Patterns` operation does not specify the order in which the User Pattern URIs appear in the response body, and there is no guarantee that the order in the response will not change in subsequent invocations. Use the **search-order-index** property to determine a pattern's position in the search order.

Example HTTP interaction

```
GET /api/console/user-patterns?type=regular-expression HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 507. List User Patterns: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 314
{
   "user-patterns":[
      {
         "element-uri":"/api/console/user-patterns/497bf4ec-1dbf-11e4-8ceb-1c6f65065a91",
         "name":"IBM Intranet User Pattern",
         "type":"regular-expression"
      },
      {
         "element-uri":"/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91",
         "name":"Company email pattern",
         "type":"regular-expression"
      }
   ]
}
```

Figure 508. List User Patterns: Response

Get User Pattern Properties

The `Get User Pattern Properties` operation retrieves the properties of a single User Pattern object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable *{user-pattern-id}* is the element ID of the User Pattern object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the User Pattern object as defined in the "Data model" on page 942. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single User Pattern object specified by *{user-pattern-id}*.

On successful execution, all of the current properties as defined in the data model for the User Pattern object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing User Pattern object and the API user must have user-related-access permission to it or action/task permission to the **Manage User Patterns** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the User Pattern object specified in the request URI or action/task permission to the **Manage User Patterns** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 948.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/user-patterns/3f6d2368-6fc5-11eb-9017-fa163e9c9075 HTTP/1.1
x-api-session: 1a2acx90ctqzmv9n391obkcg82m4pyqqw7jfrp7ylhb9is8ns1
```

Figure 509. Get User Pattern Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 15 Feb 2021 19:39:14 GMT
content-type: application/json;charset=UTF-8
content-length: 931
{
   "class":"user-pattern",
   "description":"User Pattern to match company email addresses",
   "domain-name-restrictions":null,
   "domain-name-restrictions-ldap-server-definition-uri":null,
   "element-id":"3f6d2368-6fc5-11eb-9017-fa163e9c9075",
   "element-uri":"/api/console/user-patterns/3f6d2368-6fc5-11eb-9017-fa163e9c9075",
   "ldap-group-default-template-uri":null,
   "ldap-group-ldap-server-definition-uri":null,
   "ldap-group-to-template-mappings":null,
   "ldap-server-definition-uri":null,
   "name":"Company email pattern",
   "parent":"/api/console",
   "pattern":"*@example.com",
   "replication-overwrite-possible":false,
   "retention-time":90,
   "search-order-index":11,
   "specific-template-uri":"/api/users/05339524-6bd3-11eb-a246-fa163e9c9075",
   "template-name-override":null,
   "template-name-override-default-template-uri":null,
   "template-name-override-ldap-server-definition-uri":null,
   "type":"glob-like",
   "user-template-uri":"/api/users/05339524-6bd3-11eb-a246-fa163e9c9075"
}
```

Figure 510. Get User Pattern Properties: Response

Update User Pattern Properties

The `Update User Pattern Properties` operation updates the properties of a single User Pattern object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable *{user-pattern-id}* is the element ID of the User Pattern object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the User Pattern object specified by *{user-pattern-id}* .

The URI path must designate an existing User Pattern object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Pattern object or action/task permission to the **Manage User Patterns** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated User Pattern object but not action/task permission to the **Manage User Patterns** task, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating

the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., attempting to set **template-name-override** when the **ldap-server-definition-uri** value is **null**).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **ldap-server-definition-uri** is changed from non-null to **null**, and **template-name-override** is not defined in the request body, **template-name-override** will be defaulted to **null**).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 475. Update User Pattern Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Table 475. Update User Pattern Properties: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
404 (Not Found)	1	The API user does not have the required permission for this operation.
	324	The **ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	326	The **user-template-uri** field in the request body does not designate an existing template type User object.
	343	The **template-name-override-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	344	The **ldap-group-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	345	The **domain-name-restrictions-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	346	The **specific-template-uri** field in the request body does not designate an existing template type User object.
	347	The **ldap-group-default-template-uri** field in the request body does not designate an existing template type User object.
	348	The **template-name-override-default-template-uri** field in the request body does not designate an existing template type User object.
	349	The **template-uri** field in a nested group-to-template-mapping object in the request body does not designate an existing template type User object.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 95
{
   "description":"A new and improved description of this User Pattern",
   "retention-time":30
}
```

Figure 511. Update User Pattern Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 512. Update User Pattern Properties: Response

Create User Pattern

The `Create User Pattern` operation creates a User Pattern object with the given properties on the console.

HTTP method and URI

```
POST /api/console/user-patterns
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the User Pattern's **name** property.
`description`	String	Optional	The value to be set as the User Pattern's **description** property.
`pattern`	String	Required	The value to be set as the User Pattern's **pattern** property.
`type`	String Enum	Required	The value to be set as the User Pattern's **type** property.
`retention-time`	Integer	Required	The value to be set as the User Pattern's **retention-time** property.
`user-template-uri`	String/URI	Optional	The value to be set as the User Pattern's **user-template-uri** property.
`ldap-server-definition-uri`	String/URI	Optional	The value to be set as the User Pattern's **ldap-server-definition-uri** property.
`template-name-override`	String	Optional	The value to be set as the User Pattern's **template-name-override** property.
`domain-name-restrictions`	String	Optional	The value to be set as the User Pattern's **domain-name-restrictions** property.
`specific-template-uri`	String/URI	Optional	The value to be set as the User Pattern's **specific-template-uri** property.

Field name	Type	Rqd/Opt	Description
`template-name-override-ldap-server-definition-uri`	String/URI	Optional	The value to be set as the User Pattern's **template-name-override-ldap-server-definition-uri** property.
`template-name-override-default-template-uri`	String/URI	Optional	The value to be set as the User Pattern's **template-name-override-default-template-uri** property.
`ldap-group-to-template-mappings`	Array of group-to-template-mapping objects	Optional	The value to be set as the User Pattern's **ldap-group-to-template-mappings** property.
`ldap-group-ldap-server-definition-uri`	String/URI	Optional	The value to be set as the User Pattern's **ldap-group-ldap-server-definition-uri** property.
`ldap-group-default-template-uri`	String/URI	Optional	The value to be set as the User Pattern's **ldap-group-default-template-uri** property.
`domain-name-restrictions-ldap-server-definition-uri`	String/URI	Optional	The value to be set as the User Pattern's **domain-name-restrictions-ldap-server-definition-uri** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path of the new User Pattern object.

Description

This operation creates a new User Pattern.

On successful execution of this operation the User Pattern is created using the inputs as specified by the request body. The new User Pattern is placed at the end of the pattern search order used during logon processing. The URI of the new User Pattern is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the "Request body contents" on page 952. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **template-name-override** when the **ldap-server-definition-uri** value is **null**) or the specified pattern is not unique. If a URI in the request body does not designate an existing resource of the appropriate type, status code 404 (Not Found) is

returned In addition, the API user must have action/task permission to the **Manage User Patterns** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 953, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 476. Create User Pattern: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A User Pattern with the pattern specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	324	The **ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	326	The **user-template-uri** field in the request body does not designate an existing template type User object.
	343	The **template-name-override-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	344	The **ldap-group-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	345	The **domain-name-restrictions-ldap-server-definition-uri** field in the request body does not designate an existing LDAP Server Definition object.
	346	The **specific-template-uri** field in the request body does not designate an existing template type User object.
	347	The **ldap-group-default-template-uri** field in the request body does not designate an existing template type User object.
	348	The **template-name-override-default-template-uri** field in the request body does not designate an existing template type User object.
	349	The **template-uri** field in a nested group-to-template-mapping object in the request body does not designate an existing template type User object.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Usage notes

User Patterns are searched in a defined order during logon processing. That order can be customized through the Console object's `Reorder User Patterns` operation.

Example HTTP interaction

```
POST /api/console/user-patterns HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 260
{
   "description":"User Pattern based on company email addresses",
   "name":"Company email pattern",
   "pattern":".*@our\\.company\\.com",
   "retention-time":8,
   "type":"regular-expression",
   "user-template-uri":"/api/users/ec473e56-4a7a-11e4-91ee-1c6f65065a91"
}
```

Figure 513. Create User Pattern: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 83
{
   "element-uri":"/api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 514. Create User Pattern: Response

Delete User Pattern

The `Delete User Pattern` operation deletes a User Pattern object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/user-patterns/{user-pattern-id}
```

In this request, the URI variable *{user-pattern-id}* is the element ID of the User Pattern object to be deleted.

Description

This operation removes a specified User Pattern from the console. The User Pattern is identified by the *{user-pattern-id}* variable in the URI.

Upon successfully removing the User Pattern, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing User Pattern object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated User Pattern object or action/task permission to the **Manage User Patterns** task, status code 404 (Not Found) is returned.

If the user has user-related-access permission to the designated User Pattern object but not action/task permission to the **Manage User Patterns** task, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage User Patterns** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage Note

It is permitted to delete a User Pattern even if there is a pattern-based user based on that pattern logged onto the HMC at the time of the deletion. Note that this will cause certain operations issued with the value of the **user-pattern-uri** property in that user's User object to fail, most likely with a 404 (Not Found) status code.

Example HTTP interaction

```
DELETE /api/console/user-patterns/ec5b012a-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 515. Delete User Pattern: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 516. Delete User Pattern: Response

Inventory service data

Information about the User Patterns managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for User Pattern objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular User Pattern is included only if the API user has access permission to that object as described in the `Get User Pattern Properties` operation.

For each User Pattern object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get User Pattern Properties` operation. That is, the data provided is the same as would be provided if a `Get User Pattern Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single User Pattern. This object would appear as one array entry in the response array:

```
{
   "class":"user-pattern",
   "description":"User Pattern to match company email addresses",
   "domain-name-restrictions":null,
   "domain-name-restrictions-ldap-server-definition-uri":null,
   "element-id":"3f6d2368-6fc5-11eb-9017-fa163e9c9075",
   "element-uri":"/api/console/user-patterns/3f6d2368-6fc5-11eb-9017-fa163e9c9075",
   "ldap-group-default-template-uri":null,
   "ldap-group-ldap-server-definition-uri":null,
   "ldap-group-to-template-mappings":null,
   "ldap-server-definition-uri":null,
   "name":"Company email pattern",
   "parent":"/api/console",
   "pattern":"*@example.com",
   "replication-overwrite-possible":false,
   "retention-time":90,
   "search-order-index":11,
   "specific-template-uri":"/api/users/05339524-6bd3-11eb-a246-fa163e9c9075",
   "template-name-override":null,
   "template-name-override-default-template-uri":null,
   "template-name-override-ldap-server-definition-uri":null,
   "type":"glob-like",
   "user-template-uri":"/api/users/05339524-6bd3-11eb-a246-fa163e9c9075"
}
```

Figure 517. User Pattern object: Sample inventory data

Password Rule object

A Password Rule object is an element of the console object and represents a rule which a console user(s) must follow when creating a console logon password. Each console user using local authentication is assigned a password rule. There are certain system-defined password rules available for use.

Through user-related-access permission described in "User-related-access permission" on page 892, API users are permitted to see certain Password Rule objects in a `List Password Rules` response and issue `Get Password Rule Properties` for those Password Rule objects. An API user with action/task permission to the `Manage Password Rules` task is permitted to view any Password Rule object and change any user-defined Password Rule object.

System-defined password rules

Unlike user-defined password rules, the system-defined password rules may not be modified. While system-defined password rules can be deleted and their name reused for a user-defined password rule, that practice is discouraged due to the likely confusion such a situation would cause. The names of the typical system-defined password rules include:

- Basic
- Standard
- Strict

Password rule parts

Password rule parts are optional requirements to be applied to individual parts of a password. These requirements are applied, in order, to the password, from left to right. Each of these requirements must be met by some part of the password in order for the password to meet all of the requirements of the Password Rule.

For example, to require a password to consist of 1-3 letters followed by a 4 or 5 digit number, two rule parts are defined. The first rule part requires from 1 to 3 characters, each of which must be alphabetic; the second rule part requires from 4 to 5 characters, each of which must be numeric. Passwords such as "pa1600" and "Hey90210" meet the requirements of both of those rule parts.

Data model

The Password Rule object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 98.

Table 477. Password Rule object: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for a Password Rule object is of the form `/api/console/password-rules/{password-rule-id}`, where *{password-rule-id}* is the value of the **element-id** property of the Password Rule object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of a Password Rule object is **"password-rule"**.
name	(ro)	String	The name of the Password Rule object. This name must be unique among all password rules on the console. While preexisting Password Rule names are virtually unrestricted in terms of length and characters, new Password Rule names must conform to the length and character requirements of the **name** property described in the “Base managed object properties schema” on page 100. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new Password Rule object.
description	(w)	String (0-1024)	The description of the Password Rule object. Default: an empty string

Table 477. Password Rule object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
type	—	String Enum	Identifies the type of password rule. It must be one of the following values: • **"system-defined"** - A password rule defined by the system. System-defined rules may not be modified. • **"user-defined"** - A password rule defined by a user.
expiration	(w)	Integer	The total number of days a password is valid before it expires. A value of 0 indicates that the password never expires. Default: 0
min-length	(w)	Integer (1-256)	The minimum required length of the password. Cannot be greater than **max-length**. Default: 8
max-length	(w)	Integer (1-256)	The maximum allowed length of the password. Cannot be less than **min-length**. Default: 256
consecutive-characters	(w)	Integer	The maximum number of characters that are allowed to be repeated in a row. A value of 0 indicates that there is no such limit. Default: 0
similarity-count	(w)	Integer	The maximum number of consecutive characters in the current password that can match consecutive characters in the previous password. A value of 0 indicates that there is no such limit. Default: 0
history-count	(w)	Integer	The number of previous passwords to which a new password is compared for uniqueness. A value of 0 indicates that there is no such comparison. Default: 0
case-sensitive	(w)	Boolean	Indicates whether the password is case sensitive. Default: **false**
character-rules	(w)	Array of objects	Optional rules to be applied to individual parts of the password. These rules are applied, in order, to the password, from left to right. Each of these rules must be met by some part of the password in order for the password to meet the requirements of this Password Rule. This property is an array of nested character-rule objects as described in the next table. If there are no rule parts, an empty array is provided. Default: <empty array>
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

Each nested password-rule-part object contains the following fields:

Table 478. character-rule object properties		
Name	**Type**	**Description**
min-characters	Integer	The minimum number of characters required by this password rule part. Must be at least 1, and cannot be greater than **max-characters**.
max-characters	Integer	The maximum number of characters allowed by this password rule part. Must be at least 1, and cannot be less than **min-characters**.
alphabetic	String Enum	This field determines the inclusion of alphabetic characters within this part of the password. It must be one of the following values: • **"allowed"** - There can be alphabetic characters. • **"not-allowed"** - There cannot be alphabetic characters. • **"required"** - There must be alphabetic characters.
numeric	String Enum	This field determines the inclusion of numeric characters within this part of the password. It must be one of the following values: • **"allowed"** - There can be numeric characters. • **"not-allowed"** - There cannot be numeric characters. • **"required"** - There must be numeric characters.
special	String Enum	This field determines the inclusion of special characters within this part of the password. It must be one of the following values: • **"allowed"** - There can be special characters. • **"not-allowed"** - There cannot be special characters. • **"required"** - There must be special characters. Special characters include: greater than (>), less than (<), tilde (~), exclamation mark (!), at sign (@), number sign (#), question mark (?), dollar sign ($), vertical bar (|), percent sign (%), caret (^), ampersand (&), asterisk (*), left and right parentheses ( ), underscore (_), plus sign (+), hyphen (-), equals sign (=), left and right curly braces ({ }), left and right square brackets ([ ]), back slash (\), forward slash (/), period (.), comma (,), colon (:), accent (`), quotation mark ("), semicolon (;), and apostrophe (')
custom-character-sets	Array of objects	Optional specific character requirements for this password part, as specified in an array of nested custom-character-set objects defined in the next table. This allows the specification of custom character sets and their inclusion requirement. There can be up to 2 custom character sets for a rule part. If none are defined, an empty array is provided.

Each nested specific-property object contains the following fields:

Table 479. custom-character-set object properties		
Name	**Type**	**Description**
character-set	String	A string consisting of the characters that comprise this custom character set.

Table 479. custom-character-set object properties (continued)		
Name	**Type**	**Description**
inclusion	String Enum	This field determines the inclusion of characters in this character set within part of the password. It must be one of the following values: • **"allowed"** - Characters can be included. • **"not-allowed"** - Characters cannot be included. • **"required"** - At least one character must be included.

List Password Rules

The `List Password Rules` operation lists Password Rules defined to the console.

HTTP method and URI

```
GET /api/console/password-rules
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid **type** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`password-rules`	Array of objects	Array of nested password-rule-info objects as described in the next table. If no Password Rules are to be returned, an empty array is provided.

Each nested password-rule-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The **element-uri** property of the Password Rule object.
`name`	String	The **name** property of the Password Rule object.
`type`	String Enum	The **type** property of the Password Rule object.

Description

This operation lists Password Rules defined to the console. Some basic properties are provided for each Password Rule.

If the **name** query parameter is specified the returned list is limited to those Password Rules that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

If the **type** query parameter is specified, the parameter is validated to ensure it is a valid Password Rule **type** property value. If the value is not valid, status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those Password Rules that have a **type** property matching the specified value. If the **type** parameter is omitted, this filtering is not performed.

A Password Rule is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage Password Rules** task. If there is a console Password Rule to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no Password Rules defined to the console or if no Password Rules are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- User-related-access permission to the Password Rules objects included in the response body or action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 961.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/password-rules?type=system-defined HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 518. List Password Rules: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 390
{
   "password-rules":[
      {
         "element-uri":"/api/console/password-rules/4a790766-3dbf-11e4-980d-1c6f65065a91",
         "name":"Basic",
         "type":"system-defined"
      },
      {
         "element-uri":"/api/console/password-rules/4a79360a-3dbf-11e4-980d-1c6f65065a91",
         "name":"Standard",
         "type":"system-defined"
      },
      {
         "element-uri":"/api/console/password-rules/4a792b24-3dbf-11e4-980d-1c6f65065a91",
         "name":"Strict",
         "type":"system-defined"
      }
   ]
}
```

Figure 519. List Password Rules: Response

Get Password Rule Properties

The `Get Password Rule Properties` operation retrieves the properties of a single Password Rule object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/password-rules/{password-rule-id}
```

In this request, the URI variable *{password-rule-id}* is the element ID of the Password Rule object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the Password Rule object as defined in the "Data model" on page 958. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single Password Rule object that is designated by *{password-rule-id}*.

On successful execution, all of the current properties as defined in the data model for the Password Rule object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing Password Rule object and the API user must have user-related-access permission to it or action/task permission to the **Manage Password Rules** task. If these conditions are not met, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the Password Rule object specified in the request URI or action/task permission to the **Manage Password Rules** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 963.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type, or designates a resource for which the API user does not have the required authorization.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 520. Get Password Rule Properties: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 656
{
   "case-sensitive":true,
   "character-rules":[
      {
         "alphabetic":"not-allowed",
         "custom-character-sets":[
            {
               "character-set":"*!^",
               "inclusion":"not-allowed"
            }
         ],
         "max-characters":256,
         "min-characters":8,
         "numeric":"not-allowed",
         "special":"required"
      }
   ],
   "class":"password-rule",
   "consecutive-characters":0,
   "description":"Password must be very special",
   "element-id":"ecb26fb4-4a7a-11e4-affa-1c6f65065a91",
   "element-uri":"/api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91",
   "expiration":0,
   "history-count":0,
   "max-length":256,
   "min-length":8,
   "name":"All specials",
   "parent":"/api/console",
   "replication-overwrite-possible":false,
   "similarity-count":0,
   "type":"user-defined"
}
```

Figure 521. Get Password Rule Properties: Response

Update Password Rule Properties

The `Update Password Rule Properties` operation updates the properties of a single user-defined Password Rule object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/password-rules/{password-rule-id}
```

In this request, the URI variable *{password-rule-id}* is the element ID of the Password Rule object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the Password Rule object specified by *{password-rule-id}* .

The URI path must designate an existing Password Rule object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated Password Rule object or action/task permission to the **Manage Password Rules** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated Password Rule object but not action/task permission to the **Manage Password Rules** task, status code 403 (Forbidden) is returned

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. An attempt to update a system-defined password rule is not valid and fails with status code 400 (Bad Request).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 480. Update Password Rule Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	314	This operation is not supported for an object of this type. System-defined password rules may not be updated.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 109
{
   "description":"A new and improved description of this Password Rule",
   "expiration":90,
   "history-count":5
}
```

Figure 522. Update Password Rule Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 523. Update Password Rule Properties: Response

Create Password Rule

The `Create Password Rule` operation creates a user-defined Password Rule object with the given properties.

HTTP method and URI

```
POST /api/console/password-rules
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the Password Rule's **name** property.
`description`	String	Optional	The value to be set as the Password Rule's **description** property.
`expiration`	Integer	Optional	The value to be set as the Password Rule's **expiration** property.
`min-length`	Integer	Optional	The value to be set as the Password Rule's **min-length** property.
`max-length`	Integer	Optional	The value to be set as the Password Rule's **max-length** property.
`consecutive-characters`	Integer	Optional	The value to be set as the Password Rule's **consecutive-characters** property.
`similarity-count`	Integer	Optional	The value to be set as the Password Rule's **similarity-count** property.
`history-count`	Integer	Optional	The value to be set as the Password Rule's **history-count** property.
`case-sensitive`	Boolean	Optional	The value to be set as the Password Rule's **case-sensitive** property.
`character-rules`	Array of objects	Optional	The value to be set as the Password Rule's **character-rules** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path of the new Password Rule object.

Description

This operation creates a new user-defined Password Rule.

On successful execution of this operation the Password Rule is created using the inputs as specified by the request body. The URI of the new Password Rule is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the "Request body contents" on page 967. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. If the specified name is not unique, status code 400 (Bad Request) is returned. In addition, the API user must have action/task permission to the Manage Password Rules task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 967 and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 481. Create Password Rule: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	A password rule with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/password-rules HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 298
{
   "character-rules":[
      {
         "alphabetic":"not-allowed",
         "custom-character-sets":[
            {
               "character-set":"*!^",
               "inclusion":"not-allowed"
            }
         ],
         "max-characters":256,
         "min-characters":8,
         "numeric":"not-allowed",
         "special":"required"
      }
   ],
   "description":"Password must be very special",
   "name":"All specials"
}
```

Figure 524. Create Password Rule: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 82
{
   "element-uri":"/api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91"
}
```

Figure 525. Create Password Rule: Response

Delete Password Rule

The `Delete Password Rule` operation deletes a Password Rule object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/password-rules/{password-rule-id}
```

In this request, the URI variable *{password-rule-id}* is the element ID of the Password Rule object to be deleted.

Description

This operation removes a specified Password Rule from the console. The Password Rule is identified by the *{password-rule-id}* variable in the URI.

Upon successfully removing the Password Rule, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing Password Rule object; otherwise, status code 404 (Not Found) is returned. If the user does not have user-related-access permission to the designated Password Rule object or action/task permission to the **Manage Password Rules** task, status code 404 (Not Found) is returned. If the user has user-related-access permission to the designated Password Rule object but not

action/task permission to the **Manage Password Rules** task, status code 403 (Forbidden) is returned. If any user has the specified Password Rule, the request fails and status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Password Rules** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this Password Rule.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/console/password-rules/ecb26fb4-4a7a-11e4-affa-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 526. Delete Password Rule: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT

<No response body>
```

Figure 527. Delete Password Rule: Response

Inventory service data

Information about the Password Rules managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for Password Rule objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular

Password Rule is included only if the API user has access permission to that object as described in the `Get Password Rule Properties` operation.

For each Password Rule object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get Password Rule Properties` operation. That is, the data provided is the same as would be provided if a `Get Password Rule Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single Password Rule. This object would appear as one array entry in the response array:

```
{
   "case-sensitive": false,
   "character-rules": [
      {
         "alphabetic": "required",
         "custom-character-sets": [],
         "max-characters": 1,
         "min-characters": 1,
         "numeric": "not-allowed",
         "special": "not-allowed"
      },
      {
         "alphabetic": "allowed",
         "custom-character-sets": [],
         "max-characters": 6,
         "min-characters": 4,
         "numeric": "required",
         "special": "not-allowed"
      },
      {
         "alphabetic": "required",
         "custom-character-sets": [],
         "max-characters": 1,
         "min-characters": 1,
         "numeric": "not-allowed",
         "special": "not-allowed"
      }
   ],
   "class": "password-rule",
   "consecutive-characters": 2,
   "description": "",
   "element-id": "56d11882-eaff-11e2-9ec7-5cf3fcae8019",
   "element-uri": "/api/console/password-rules/56d11882-eaff-11e2-
      9ec7-5cf3fcae8019",
   "expiration": 180,
   "history-count": 0,
   "max-length": 8,
   "min-length": 6,
   "name": "Strict",
   "parent": "/api/console",
   "replication-overwrite-possible": false,
   "similarity-count": 0,
   "type": "system-defined"
}
```

Figure 528. Password Rule object: Sample inventory data

LDAP Server Definition object

An LDAP Server Definition object is an element of the console object and contains information about an LDAP server that may be used for console user authorization purposes. LDAP servers are sometimes referred to as Enterprise Directory Servers on the zManager user interface and publications.

All API users are permitted to issue `List LDAP Server Definitions` and retrieve very basic information for the LDAP Server Definition, if any, that applies to them. That is, they can use that operation

to retrieve very basic information for the LDAP Server Definition identified in their User object. An API user with action/task permission to the **Manage LDAP Server Definitions** task is permitted to view and change any LDAP Server Definition object.

Data model

This object contains the following properties. Certain properties are only valid when mutable prerequisite properties have specific values. When such properties are not valid, their value is **null**. For instance the **search-filter** is **null** when the **location-method** value is **"pattern"**.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

Table 482. LDAP Server Definition object: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for an LDAP Server Definition object is of the form `/api/console/ldap-server-definitions/{ldap-server-definition-id}`, where *{ldap-server-definition-id}* is the value of the **element-id** property of the LDAP Server Definition object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the console object.
class	—	String	The class of an LDAP Server Definition object is **"ldap-server-definition"**.
name	(ro)	String	The name of the LDAP Server Definition object. This name must be unique among all LDAP Server Definition objects defined to the console. While preexisting LDAP Server Definition names are virtually unrestricted in terms of length and characters, new LDAP Server Definition names must conform to the length and character requirements of the **name** property described in the "Base managed object properties schema" on page 100. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new LDAP Server Definition object.
description	(w)	String (0-1024)	The description of the LDAP Server Definition object. Default: an empty string
primary-hostname-ipaddr	(w)	String, String/ IPV4 address, or String/IPV6 address	The host name or IP address of the primary LDAP server. It must contain at least 1 non-whitespace character.

Table 482. LDAP Server Definition object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
connection-port	(w)	Integer	The TCP port number, which must be greater than 0, on which the server accepts connections, or **null** if the server uses the standard LDAP port appropriate for the value of the **use-ssl** property. The standard LDAP port values are 636 if **use-ssl** is **true** and 389 if **use-ssl** is **false** Default: **null**
backup-hostname-ipaddr	(w)	String, String/IPV4 address, or String/IPV6 address	The host name or IP address of the backup LDAP server (which must contain at least 1 non-whitespace character), or **null** if there is none. Default: **null**
use-ssl	(w)	Boolean	Indicates whether the server uses SSL for incoming connections. Default: **false**
tolerate-untrusted-certificates	(w)	Boolean	Indicates whether the server should tolerate self-signed or otherwise untrusted certificates. Prerequisite: **use-ssl** is **true** Default: **false**
authentication-type[1,2]	(w)	String Enum	The type of authentication to use with the LDAP server when locating a user's LDAP directory entry or determining a user's group membership. Must be one of the following: • **"anonymous"** - Perform LDAP searches with no credentials. The LDAP server must be configured to allow this. **bind-distinguished-name** and **bind-password** must be null. • **"simple"** - Perform LDAP searches using the credentials provided in **bind-distinguished-name** and **bind-password**, both of which must be non-empty strings. • **"direct"** - Perform LDAP compares using the user's credentials and **user-dn-placeholder**, **group-dn-placeholder**, and **group-membership-attribute** when determining a user's group membership. Direct authentication is intended for use in LDAP Server Definitions that are used for determining a user's group memberships, but it can also be used when locating a user's LDAP directory entry. Default: **"anonymous"** if **bind-distinguished-name** and **bind-password** are null; otherwise, **"simple"**. [Added by feature **ldap-direct-authentication**]

Table 482. LDAP Server Definition object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
bind-distinguished-name[1]	(w)	String	The distinguished name to use on the bind for the initial connection. This is only required when **authentication-type** is **"simple"**, in which case it must contain at least 1 non-whitespace character. It must be null when **authentication-type** is **"anonymous"**. [Updated by feature **ldap-direct-authentication**] Default: **null** Prerequisite: Both **bind-distinguished-name** and **bind-password** must be **null**, or they both must be non-empty strings.
bind-password[1]	(wo)	String	The password to use on the bind for the initial connection. This is only required when **authentication-type** is "**"simple**, in which case it must contain at least 1 non-whitespace character. It must be null when **authentication-type** is **"anonymous"**. [Updated by feature **ldap-direct-authentication**] Default: **null** Prerequisite: Both **bind-distinguished-name** and **bind-password** must be **null**, or they both must be non-empty strings.
location-method	(w)	String Enum	The method this server uses to locate a user's directory entry. Must be one of the following: • **"pattern"** - Use a distinguished name pattern. • **"subtree"** - Use a distinguished name subtree. Default: **"pattern"** Not used when **authentication-type** is **"direct"**. [Updated by feature **ldap-direct-authentication**]

Table 482. LDAP Server Definition object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
search-distinguished-name	(w)	String	The distinguished name to use when searching for a user's directory entry. If **location-method** is **"pattern"**, this is the distinguished name pattern to use when searching for a user's directory entry. It must include the string "{0}", which indicates where in the pattern the user ID is to be substituted. The user ID is the value of the **userid-on-ldap-server** property of the user's User object, unless it is **null**, in which case the **name** property of the User object is used. If **location-method** is **"subtree"**, this is the distinguished name of the subtree to search for a user's directory entry. Not used when **authentication-type** is **"direct"**. [Updated by feature **ldap-direct-authentication**]
search-scope	(w)	String Enum	Indicates how much of the subtree should be searched when searching for a user's directory entry in a subtree. The filter is specified in the **search-filter** property. Must be one of the following: • **"all"** - Search the entire subtree. • **"one-level"** - Search one level only. Prerequisite: **location-method** is **"subtree"** Default: **"all"** Not used when **authentication-type** is **"direct"**. [Updated by feature **ldap-direct-authentication**]
search-filter	(w)	String	The LDAP search filter to use when searching for a user's directory entry in a subtree. It must include the string "{0}", which indicates where in the filter the user ID is to be substituted. The user ID is the value of the **userid-on-ldap-server** property of the user's User object, unless it is **null**, in which case the **name** property of the User object is used. The **search-scope** property specifies how this filter is used during the search. Prerequisite: **location-method** is **"subtree"** Not used when **authentication-type** is **"direct"**. [Updated by feature **ldap-direct-authentication**]

Table 482. LDAP Server Definition object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
user-dn-placehoder	(w)	String	This specifies the format for the LDAP distinguished name of a user. This is used when locating the user's LDAP directory entry during logon processing and **"direct"** authentication with the LDAP server is specified. When **authentication-type** is **"direct"**, it must include the string "{0}", which indicates where in the placeholder the user ID is to be substituted; otherwise, it must be null. Default: **null** [Added by feature **ldap-direct-authentication**]
group-dn-placeholder	(w)	String	This specifies the format for the LDAP distinguished name of a group. This is used when determining if the user is a member of a specific group and **"direct"** authentication with the LDAP server is specified. When **authentication-type** is **"direct"**, it must include the string "{0}", which indicates where in the placeholder the group name is to be substituted; otherwise, it must be null. Default: **null** [Added by feature **ldap-direct-authentication**]
group-membership-attribute	(w)	String	The name of the group membership attribute in the user's directory entry. This is used when determining if the user is a member of a specific group and **"direct"** authentication with the LDAP server is specified. When **authentication-type** is **"direct"**, it must be a non-empty string; otherwise, it must be null. Default: **null** [Added by feature **ldap-direct-authentication**]
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

[1]If a Create LDAP Server Definition or Update LDAP Server Definition Properties operation sets **bind-distinguished-name** and **bind-password** to null, and does not include **authentication-type**, then **authentication-type** is automatically set to **"anonymous"**. If a Create LDAP Server Definition or Update LDAP Server Definition Properties operation sets **bind-distinguished-name** and **bind-password** to non-empty strings, and does not include **authentication-type**, then **authentication-type** is automatically set to **"simple"**.

[2]An LDAP Server Definition's **authentication-type** property cannot be changed to **"direct"** if it is specified in any of the following User Pattern properties: **ldap-server-definition-uri**, **template-name-override-ldap-server-definition-uri**, **domain-name-restrictions-ldap-server-definition-uri**.

List LDAP Server Definitions

The `List LDAP Server Definitions` operation lists LDAP Server Definitions defined to the console.

HTTP method and URI

```
GET /api/console/ldap-server-definitions
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
ldap-server-definitions	Array of objects	Array of nested ldap-server-definition-info objects as described in the next table.

Each nested ldap-server-definition-info object contains the following fields:

Field name	Type	Description
element-uri	String/URI	The **element-uri** property of the LDAP Server Definition object.
name	String	The **name** property of the LDAP Server Definition object.

Description

This operation lists LDAP Server Definitions defined to the console. Some basic properties are provided for each LDAP Server Definition.

If the **name** query parameter is specified the returned list is limited to those LDAP Server Definitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not performed.

An LDAP Server Definition is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage LDAP Server Definitions** task. If there is an LDAP Server Definition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no LDAP Server Definitions defined to the console or if no LDAP Server Definitions are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the LDAP Server Definition objects included in the response body or action/task permission to the **Manage LDAP Server Definitions** task

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 977.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/ldap-server-definitions?name=IBM.*&name=Company.* HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 529. List LDAP Server Definitions: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT
content-type: application/json;charset=UTF-8
content-length: 276
{
   "ldap-server-definitions":[
      {
         "element-uri":"/api/console/ldap-server-definitions/
            3ac6550e-1dbb-11e4-9aa4-1c6f65065a91",
         "name":"IBM LDAP server"
      },
      {
         "element-uri":"/api/console/ldap-server-definitions/
            ece481ca-4a7a-11e4-8777-1c6f65065a91",
         "name":"Company LDAP server"
      }
   ]
}
```

Figure 530. List LDAP Server Definitions: Response

Get LDAP Server Definition Properties

The `Get LDAP Server Definition Properties` operation retrieves the properties of a single LDAP Server Definition object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/ldap-server-definitions/{ldap-server-definition-id}
```

In this request, the URI variable *{ldap-server-definition-id}* is the element ID of the LDAP Server Definition object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the LDAP Server Definition object as defined in the "Data model" on page 972. Field

names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties of a single LDAP Server Definition object that is designated by *{ldap-server-definition-id}*.

On successful execution, all of the current properties as defined in the data model for the LDAP Server Definition object, except those designated as write-only properties, are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 978.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/ldap-server-definitions/513ee63a-4d03-11ee-b858-fa163ed14f64 HTTP/1.1
x-api-session: 471ufmxle535ade16uqbgnhh94z8q89y96kj3pft9b1shjsbqu
```

Figure 531. Get LDAP Server Definition Properties: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Wed, 06 Sep 2023 22:19:51 GMT
Content-Type: application/json;charset=UTF-8
Content-Length: 770
{
   "authentication-type":"anonymous",
   "backup-hostname-ipaddr":null,
   "bind-distinguished-name":null,
   "class":"ldap-server-definition",
   "connection-port":636,
   "description":"Main directory server for the company",
   "element-id":"513ee63a-4d03-11ee-b858-fa163ed14f64",
   "element-uri":"/api/console/ldap-server-definitions/513ee63a-4d03-11ee-b858-fa163ed14f64",
   "group-dn-placeholder":null,
   "group-membership-attribute":null,
   "location-method":"subtree",
   "name":"Company Directory Server 1",
   "parent":"/api/console",
   "primary-hostname-ipaddr":"ldap1.example.com",
   "replication-overwrite-possible":false,
   "search-distinguished-name":"ou=ourcompany,o=example.com",
   "search-filter":"mail={0}",
   "search-scope":"all",
   "tolerate-untrusted-certificates":false,
   "use-ssl":true,
   "user-dn-placeholder":null
}
```

Figure 532. Get LDAP Server Definition Properties: Response

Update LDAP Server Definition Properties

The `Update LDAP Server Definition Properties` operation updates the properties of a single LDAP Server Definition object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/ldap-server-definitions/{ldap-server-definition-id}
```

In this request, the URI variable *{ldap-server-definition-id}* is the element ID of the LDAP Server Definition object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the data model for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the LDAP Server Definition object specified by *{ldap-server-definition-id}*.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in the request body contents section. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid

because a prerequisite is not met (e.g., attempting to set **search-filter** when the **location-method** value is **"pattern"**).

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided and no prerequisite property is changed remain unchanged by this operation. A property's value is set to its default value if the field is not included in the request body and a prerequisite field is changed such that the prerequisite condition becomes satisfied (e.g., if **location-method** is changed from **"pattern"** to **"subtree"**, and **search-scope** is not defined in the request body, **search-scope** will be defaulted to **all**).

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91 HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 133
{
   "backup-hostname-ipaddr":"ldap2.my.company.com",
   "description":"A new and improved description of this LDAP Server Definition"
}
```

Figure 533. Update LDAP Server Definition Properties: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT

<No response body>
```

Figure 534. Update LDAP Server Definition Properties: Response

Create LDAP Server Definition

The `Create LDAP Server Definition` operation creates an LDAP Server Definition object with the given properties.

HTTP method and URI

```
POST /api/console/ldap-server-definitions
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the LDAP Server Definition's **name** property.
`description`	String	Optional	The value to be set as the LDAP Server Definition's **description** property.
`primary-hostname-ipaddr`	String	Required	The value to be set as the LDAP Server Definition's **primary-hostname-ipaddr** property.
`connection-port`	Integer	Optional	The value to be set as the LDAP Server Definition's **connection-port** property.
`backup-hostname-ipaddr`	String	Optional	The value to be set as the LDAP Server Definition's **backup-hostname-ipaddr** property.
`use-ssl`	Boolean	Optional	The value to be set as the LDAP Server Definition's **use-ssl** property.
`tolerate-untrusted-certificates`	Boolean	Optional	The value to be set as the LDAP Server Definition's **tolerate-untrusted-certificates** property.
`bind-distinguished-name`	String	Optional	The value to be set as the LDAP Server Definition's **bind-distinguished-name** property.
`bind-password`	String	Optional	The value to be set as the LDAP Server Definition's **bind-password** property.
`location-method`	String Enum	Optional	The value to be set as the LDAP Server Definition's **location-method** property.
`search-distinguished-name`	String	Required	The value to be set as the LDAP Server Definition's **search-distinguished-name** property.
`search-scope`	String Enum	Optional	The value to be set as the LDAP Server Definition's **search-scope** property.
`search-filter`	String	Required if **location-method** is **"subtree"**	The value to be set as the LDAP Server Definition's **search-filter** property.
`authentication-type`	String Enum	Optional	The value to be set as the LDAP Server Definition's **authentication-type** property.

Field name	Type	Rqd/Opt	Description
`user-dn-placeholder`	String	Optional	The value to be set as the LDAP Server Definition's **user-dn-placeholder** property.
`group-dn-placeholder`	String	Optional	The value to be set as the LDAP Server Definition's **group-dn-placeholder** property.
`group-membership-attribute`	String	Optional	The value to be set as the LDAP Server Definition's **group-membership-attribute** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path of the new LDAP Server Definition object.

Description

This operation creates a new LDAP Server Definition.

On successful execution of this operation the LDAP Server Definition is created using the inputs as specified by the request body. The URI of the new LDAP Server Definition is provided in the response body and in a **Location** response header as well.

The request body is validated against the schema described in the "Request body contents" on page 982. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because a prerequisite is not met (e.g., specifying **search-filter** when the **location-method** value is **"pattern"**) or the specified name is not unique. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 983, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 483. Create LDAP Server Definition: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	An LDAP Server Definition with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/ldap-server-definitions HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
content-length: 264
{
   "description":"Directory server for the company",
   "location-method":"subtree",
   "name":"Company LDAP server",
   "primary-hostname-ipaddr":"ldap1.our.company.com",
   "search-distinguished-name":"o=our,ou=company.com",
   "search-filter":"email={0}",
   "use-ssl":true
}
```

Figure 535. Create LDAP Server Definition: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:33 GMT
content-type: application/json;charset=UTF-8
content-length: 94
{
   "element-uri":"/api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91"
}
```

Figure 536. Create LDAP Server Definition: Response

Delete LDAP Server Definition

The `Delete LDAP Server Definition` operation deletes an LDAP Server Definition object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/ldap-server-definitions/{ldap-server-definition-id}
```

In this request, the URI variable *{ldap-server-definition-id}* is the element ID of the LDAP Server Definition object to be deleted.

Description

This operation removes a specified LDAP Server Definition from the console. The LDAP Server Definition is identified by the *{ldap-server-definition-id}* variable in the URI.

Upon successfully removing the LDAP Server Definition, HTTP status code 204 (No Content) is returned and no response body is provided.

The URI path must designate an existing LDAP Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage LDAP Server Definitions** task; otherwise, status code 403 (Forbidden) is returned. If any user is defined to use the specified LDAP Server Definition, the request fails and HTTP status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage LDAP Server Definitions** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the correct type or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users or User Patterns has this LDAP Server Definition.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/console/ldap-server-definitions/ece481ca-4a7a-11e4-8777-1c6f65065a91
   HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 537. Delete LDAP Server Definition: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Thu, 02 Oct 2014 21:27:34 GMT

<No response body>
```

Figure 538. Delete LDAP Server Definition: Response

Inventory service data

Information about the LDAP Server Definitions managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for LDAP Server Definition objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular LDAP Server Definition is included only if the API user has access permission to that object as described in the `Get LDAP Server Definition Properties` operation.

For each LDAP Server Definition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get LDAP Server Definition Properties` operation. That is, the data provided is the same as would be provided if a `Get LDAP Server Definition Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single LDAP Server Definition. This object would appear as one array entry in the response array:

```
{
   "authentication-type":"anonymous",
   "backup-hostname-ipaddr": null,
   "bind-distinguished-name": null,
   "class": "ldap-server-definition",
   "connection-port": null,
   "description": "",
   "element-id": "ffbf71f4-370d-11e4-a5fc-5ef3fcae8020",
   "element-uri": "/api/console/ldap-server-definitions/ffbf71f4-370d-11e4-
      a5fc-5ef3fcae8020",
   "group-dn-placeholder":null,
   "group-membership-attribute":null,
   "location-method": "pattern",
   "name": "Temp_LDAP_13_56ba2f43-98c0-4848-9af8-cdb45b56f082",
   "parent": "/api/console",
   "primary-hostname-ipaddr": "bluepages.example.com",
   "replication-overwrite-possible": false,
   "search-distinguished-name": "uid={0}744,c=in,ou=bluepages,o=example.com",
   "search-filter": null,
   "search-scope": null,
   "tolerate-untrusted-certificates": null,
   "use-ssl": false
   "user-dn-placeholder":null
}
```

Figure 539. LDAP Server Definition object: Sample inventory data

MFA Server Definition object

An MFA Server Definition object is an element of the Console object and contains information about an MFA server that may be used for console user authorization purposes.

All API users are permitted to issue `List MFA Server Definitions` and retrieve very basic information for the MFA Server Definitions, if any, that apply to them. That is, they can use that operation to retrieve very basic information for the MFA Server Definitions identified in their User object. An API user with action/task permission to the **Manage Multi-factor Authentication** task is permitted to view and change any MFA Server Definition object.

Data model

The MFA Server Definition object contains the following properties.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

Table 484. MFA Server Definition object: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path for an MFA Server Definition object is of the form `/api/console/mfa-server-definitions/`*`{mfa-server-definition-id}`*, where *{mfa-server-definition-id}* is the value of the **element-id** property of the MFA Server Definition object.
element-id	—	String (36)	The unique identifier for this object.
parent	—	String/ URI	The canonical URI path of the Console object.
class	—	String (21)	The class of an MFA Server Definition object is **"mfa-server-definition"**.
name	(w)(pc)	String	The name of the MFA Server Definition object. This name must be unique among all of the console's MFA Server Definitions, and it must conform to the length and character requirements of the **name** property described in the "Base managed object properties schema" on page 100. For the purpose of verifying uniqueness, this name is treated in a case-insensitive fashion when used to create a new MFA Server Definition object or update the name of an existing one.
description	(w)(pc)	String (0-1024)	The description of the MFA Server Definition object. Default: an empty string
hostname-ipaddr	(w)(pc)	String/ Hostname, String/IPV4 Address, or String/IPV6 Address	The hostname or IP address of the server. Must not be null.

Table 484. MFA Server Definition object: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
port	(w)(pc)	Integer (1-65535)	The TCP port number on which the server accepts connections. Default: 6789
replication-overwrite-possible	—	Boolean	Indicates whether this object is customizable data that is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service.

List MFA Server Definitions

The `List MFA Server Definitions` operation lists MFA Server Definitions defined to the console.

HTTP method and URI

```
GET /api/console/mfa-server-definitions
```

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`mfa-server-definitions`	Array of mfa-server-definition-info objects	Array of nested mfa-server-definition-info objects as described in the next table.

Each nested mfa-server-definition-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/URI	The **element-uri** property of the MFA Server Definition object.
`name`	String	The **name** property of the MFA Server Definition object.

Description

This operation lists MFA Server Definitions defined to the console. Some basic properties are provided for each MFA Server Definition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those MFA Server Definitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

An MFA Server Definition is included in the list only if the API user has user-related-access permission to that object or action/task permission to the **Manage Multi-factor Authentication** task. If there is an MFA

Server Definition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no MFA Server Definitions defined to the console or if no MFA Server Definitions are to be included in the results due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- User-related-access permission to the MFA Server Definition objects included in the response body or action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 988.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter contains an invalid value.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/mfa-server-definitions HTTP/1.1
x-api-session: 2q7icwfrl9deu0rxrupb0p6mobcvirs0w4y1bvtfocrz0ix5fl
```

Figure 540. List MFA Server Definitions: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:13:32 GMT
content-type: application/json
content-length: 254
{
   "mfa-server-definitions":[
      {
         "element-uri":"/api/console/mfa-server-definitions/14ef02da-c857-11e9-8189-
fa163e8e8d46",
         "name":"MFA server 2"
      },
      {
         "element-uri":"/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-
fa163e8e8d46",
         "name":"MFA server 1"
      }
   ]
}
```

Figure 541. List MFA Server Definitions: Response

Get MFA Server Definition Properties

The `Get MFA Server Definition Properties` operation retrieves the properties of a single MFA Server Definition object that is designated by its element ID.

HTTP method and URI

```
GET /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object whose properties are to be returned.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties of the MFA Server Definition object as defined in the "Data model" on page 987. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation returns the current property values of the MFA Server Definition object specified by *{mfa-server-definition-id}*.

On successful execution, the current values of the properties as defined in "Data model" on page 987 for the MFA Server Definition object are provided in the response body, and HTTP status code 200 (OK) is returned.

The URI path must designate an existing MFA Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentication** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 990.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The element ID in the request URI *{mfa-server-definition-id}* does not designate an existing MFA Server Definition object or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 29t78zp0syhk4lhe5xyx6ib53qloudk7zthe6dsd5nnefcvkxa
```

Figure 542. Get MFA Server Definition Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:15:43 GMT
content-type: application/json
content-length: 364
{
   "class":"mfa-server-definition",
   "description":"Company primary IBM MFA server",
   "element-id":"3007226a-c856-11e9-9df4-fa163e8e8d46",
   "element-uri":"/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46",
   "hostname-ipaddr":"mfa1.internal.example.com",
   "name":"MFA server 1",
   "parent":"/api/console",
   "port":6789,
   "replication-overwrite-possible":false
}
```

Figure 543. Get MFA Server Definition Properties: Response

Update MFA Server Definition Properties

The `Update MFA Server Definition Properties` operation updates the properties of a single MFA Server Definition object that is designated by its element ID.

HTTP method and URI

```
POST /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object whose properties are to be updated.

Request body contents

The request body is expected to contain a JSON object that provides the new values of any writable property that is to be updated by this operation. Field names and data types in this JSON object are expected to match the corresponding property names and data types defined by the "Data model" on page 987 for this object type. The JSON object can and should omit fields for properties whose values are not to be changed by this operation.

Description

This operation updates writable properties of the MFA Server Definition object specified by *{mfa-server-definition-id}*.

On successful execution, the MFA Server Definition object has been updated with the supplied property values and status code 204 (No Content) is returned without supplying a response body.

The URI path must designate an existing MFA Server definition object; otherwise, HTTP status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentication** task; otherwise, HTTP status code 403 (Forbidden) is returned.

The request body is validated against the schema described in "Request body contents" on page 991. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

The request body does not need to specify a value for all writable properties, but rather can and should contain fields only for the properties to be updated. Object properties for which no input value is provided remain unchanged by this operation unless a prerequisite or linked property is changed.

If the update changes the value of any property for which Property Change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The element ID in the request URI *{mfa-server-definition-id}* does not designate an existing MFA Server Definition object or the API user has no access permission to it.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 5oqk73rga8zzqrqne7bmpzlscpppu1onkno12jbgt2q095fjvw
content-type: application/json
content-length: 14
{
   "port":6790
}
```

Figure 544. Update MFA Server Definition Properties: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:18:22 GMT
content-type: application/json

<No response body>
```

Figure 545. Update MFA Server Definition Properties: Response

Create MFA Server Definition

The `Create MFA Server Definition` operation creates an MFA Server Definition object with the given properties.

HTTP method and URI

```
POST /api/console/mfa-server-definitions
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the MFA Server Definition's **name** property.
`description`	String	Optional	The value to be set as the MFA Server Definition's **description** property.
`hostname-ipaddr`	String/ Hostname, String/ IPV4 Address, or String/ IPV6 Address	Required	The value to be set as the MFA Server Definition's **hostname-ipaddr** property.
`port`	Integer	Optional	The value to be set as the MFA Server Definition's **port** property.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path of the new MFA Server Definition object.

Description

This operation creates a new MFA Server Definition object.

On successful execution of this operation the MFA Server Definition is created using the values specified in the request body. The URI of the new MFA Server Definition is provided in the response body and in a **Location** response header.

The request body is validated against the schema described in the "Request body contents" on page 993. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. The request body validation will fail if it contains a property that is not valid because the specified name is not unique. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentiction** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 993, and the **Location** response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 485. Create MFA Server Definition: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	An MFA Server Definition with the name specified in the request body already exists.
403 (Forbidden)	1	The API user does not have the required permission for this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/console/mfa-server-definitions HTTP/1.1
x-api-session: 1l0nki6fyqq4zuue0lxmlahx8tnchoihm2w27cin0blkwma5l6
content-type: application/json
content-length: 121
{
   "description":"Company primary IBM MFA server",
   "hostname-ipaddr":"mfa1.internal.example.com",
   "name":"MFA server 1"
}
```

Figure 546. Create MFA Server Definition: Request

```
201 Created
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:06:50 GMT
content-type: application/json
content-length: 90
{
   "element-uri":"/api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46"
}
```

Figure 547. Create MFA Server Definition: Response

Delete MFA Server Definition

The `Delete MFA Server Definition` operation deletes an MFA Server Definition object designated by its element ID.

HTTP method and URI

```
DELETE /api/console/mfa-server-definitions/{mfa-server-definition-id}
```

In this request, the URI variable *{mfa-server-definition-id}* is the element ID of the MFA Server Definition object to be deleted.

Description

This operation removes the MFA Server Definition specified by *{mfa-server-definition-id}* from the console.

On successful execution, the MFA Server Definition object has been removed and no response body is provided. An Inventory Change notification is emitted asynchronously to this operation.

The URI path must designate an existing MFA Server Definition object; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have action/task permission to the **Manage Multi-factor Authentication** task; otherwise, status code 403 (Forbidden) is returned.

If any user or user template is defined to use the specified MFA Server Definition, the request fails and HTTP status code 409 (Conflict) is returned.

Authorization requirements

This operation has the following authorization requirement:

- Action/task permission to the **Manage Multi-factor Authentication** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The element ID in the request URI *{mfa-server-definition-id}* does not designate an existing MFA Server Definition object or the API user has no access permission to it.
409 (Conflict)	317	The object cannot be deleted at this time. One or more users have this MFA Server Definition.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/console/mfa-server-definitions/3007226a-c856-11e9-9df4-fa163e8e8d46 HTTP/1.1
x-api-session: 5qlow7gr99vbc4gyidnc769ipivfsh6z3z2aj54ydwh9z0sbfn
```

Figure 548. Delete MFA Server Definition: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 26 Aug 2019 23:20:28 GMT

<No response body>
```

Figure 549. Delete MFA Server Definition: Response

Inventory service data

Information about the MFA Server Definitions managed by the console can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for MFA Server Definition objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"console"** are to be included. An entry for a particular MFA Server Definition is included only if the API user has access permission to that object as described in the `Get MFA Server Definition Properties` operation.

For each MFA Server Definition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for the `Get MFA Server Definition Properties` operation. That is, the data provided is the same as would be provided if a `Get MFA Server Definition Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single MFA Server Definition. This object would appear as one array entry in the response array:

```
{
      "class":"mfa-server-definition",
      "description":"Backup MFA server",
      "element-id":"14ef02da-c857-11e9-8189-fa163e8e8d46",
      "element-uri":"/api/console/mfa-server-definitions/14ef02da-c857-11e9-8189-fa163e8e8d46",
      "hostname-ipaddr":"mfa2.internal.example.com",
      "name":"MFA server 2",
      "parent":"/api/console",
      "port":5402,
      "replication-overwrite-possible":false
   }
```

Figure 550. MFA Server Definition object: Sample inventory data

Group Object

The Hardware Management Console and Support Element each provide a fixed set of system-defined groups to which managed objects of certain types automatically belong, as members. For example, defined CPCs are automatically members of the CPC group. By their nature, the members of the

system-defined groups are obtainable through list operations of the appropriate API. For example, all the CPCs managed by a Hardware Management Console can be obtained through a List CPCs operation. Therefore, list operations for system-defined groups are unnecessary. By their nature, the existence of a system-defined group and its content (members) is implicit. Therefore, create/delete operations for system-defined groups are both unnecessary and inappropriate.

These system-defined groups are distinct from user-defined ("custom") groups. The latter are explicitly created by users for their own purposes: for example, it may be convenient for management purposes to take some proper subset of the members of the system-defined CPC group as a user-defined group of CPCs. User-defined groups may be homogeneous (all members of the same managed object type, as in this previous example), but need not be.

A Group object represents one or more managed objects which are called group members. Each member is of some object type: CPC, Logical Partition, etc. Note that groups may be heterogeneous (with member objects of differing types), and may even have other groups as members.

Users may define groups in one of two ways:

1. by use of a pattern-match expression to implicitly define membership (pattern-matching group)
2. by explicitly choosing members.

This API can be used to view/manage custom groups, and membership within these groups. The latter is subject to restrictions, based on which of the two fundamentally different means of definition the user employed:

- If pattern-matching was specified, then group membership is "implicit". In this case, operations to add/remove a member are unnecessary (simply create/delete the managed object, itself, using the appropriate API operation). Accordingly, member-management operations are not supported for groups using pattern-matching.
- If pattern-matching was not specified, then group membership is “explicit”, and in this case operations to add/remove group members are both useful and appropriate. Accordingly, for custom groups not based on pattern-matching, member-management operations are supported. Note that such operations do not affect the member object itself, only its group membership status.
- When groups are defined using pattern-matching, the types of managed objects to which pattern-matching is applied must be explicitly specified. Regardless of the POSIX regular expression specified as the match pattern, managed objects whose names match the pattern but who are not of the specified object type(s) are not considered to be members of the group.
- Groups are not intrinsically ordered in any way, nor are members within a given group. List-oriented operations therefore do not return ordered results.

Data model

This object includes the properties defined in the “Base managed object properties schema” on page 100, but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status.

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 98

The following class-specific specializations apply to the other base managed object properties:

Table 486. Group object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Group object, of the form `/api/groups/{group-id}` where *{group-id}* is the value of the **object-id** property of the Group object.
parent	—	String	This property is always a null object.
class	—	String	The class of a Group object is **"group"**.

Table 486. Group object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
name	(ro)	String	The group name specified by the user when the group was created
description	(ro)	String	The description specified by the user when the group was created, or if none was provided, the IBM provided caption text.
replication-overwrite-possible	—	Boolean	Indicates whether this object is replicated to this HMC from an HMC configured as a Data Source in the Data Replication service. **Note:** Always **false** for the Support Element.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 487. Group object: class specific additional properties		
Name	**Type**	**Description**
match-info	match-info object	A nested object which pertains to pattern-matching groups only, as described in the next table. An empty value is returned for groups which do not use pattern-matching.

The match-info object contains the following fields:

Table 488. match-info object properties		
Name	**Type**	**Description**
pattern	String	A regular expression used to define membership for pattern-matching groups. This field has no length limitations.
types	Array of String Enum	Specifies the type(s) of objects that are eligible for membership in pattern-matching groups. One or more of the following: • **"custom-groups"** - zManager API objects of class "group" • **"defined-cpc"** - zManager API objects of class "cpc" • **"logical-partition"** - zManager API objects of class "logical-partition"

List Custom Groups

The `List Custom Groups` operation lists the custom groups which are visible to the API user. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/groups
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
groups	Array of group-info objects	Array of nested objects which identify groups that are visible to the API user.

Each nested group-info object contains the following fields:

Field name	Type	Description
object-uri	String/URI	The value of the Group object's **object-uri** property.
name	String	The value of the Group object's **name** property.
target-name	String (1-17)	The **target-name** property of the CPC object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Group objects which are visible to the API user. Only groups to which the caller has authorization will be returned.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section. If no groups exist, or if no groups are visible to the API user, HTTP status code 200 (OK) is returned, along with an empty response body.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface object-access permission to the Group object.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described “Response body contents” on page 999.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/groups HTTP/1.1
x-api-session: 4ipkcgbjpy5kocelt652l3dvb85gi81iqy5bz8yrpt6vtrt8ks
```

Figure 551. List Custom Groups: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:02:42 GMT
content-type: application/json;charset=UTF-8
content-length: 283
{
   "groups": [
      {
         "name": "Finance department CPCs",
         "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341"
      },
      {
         "name": "Test Group",
         "object-uri": "/api/groups/febde5ab-a4a6-35bf-9e01-83aae59d7e52"
      }
   ]
}
```

Figure 552. List Custom Groups: Response

Get Custom Group Properties

The `Get Custom Group Properties` operation retrieves the properties of a single Group object that is designated by the *{group-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/groups/{group-id}
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Response body contents

On success, HTTP status code 200 (OK) is returned and the response body contains an object that provides the current values of the properties for the Group object as defined in "Data model" on page 997. Field names and data types in the object are the same as the property names and data types defined in the data model.

Description

This operation returns the current properties for the Group object designated by *{group-id}*.

The URI path *{group-id}* must designate an existing Group object.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface object-access permission to the Group object designated by *{group-id}*.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1000.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The **object-id** in the URI (*{group-id}*) does not designate an existing group, or the API user does not have sufficient access (as described above).

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341 HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
```

Figure 553. Get Custom Group Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:45 GMT
content-type: application/json;charset=UTF-8
content-length: 250
{
   "class": "group",
   "description": "Spacely Sprockets Web Servers",
   "replication-overwrite-possible": false,
   "is-locked": false,
   "match-info": {},
   "name": "SS-Web-Servers",
   "object-id": "ee2782af-dd98-3ec0-bc2d-cfe2e9154341",
   "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341",
   "parent": null
}
```

Figure 554. Get Custom Group Properties: Response

Create Custom Group

Use the `Create Custom Group` operation to create a custom group. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/groups
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The name for the new custom Group object
`description`	String	Optional	The description for the new custom Group object
`match-info`	match-info object	Optional	A nested object describing the pattern match. If not provided, this is not a pattern-match custom group. Refer to "Class specific additional properties" on page 998 for details.

Response body contents

Field name	Type	Description
`object-uri`	String	The object URI of the new custom group.

Description

Group objects are programmatically identified by object-id and not by name. To avoid the confusion which might result from allowing redundant names, the **name** property is required for this operation, and the (case-sensitive) value supplied for the name property must be distinct from that of all currently-existing Group objects. In keeping with restrictions imposed by the Hardware Management Console's Graphical User Interface (GUI), the following set of names is also not allowed:

- the current name of the Console
- the GUI View names {"Groups", "Exceptions", "Active Tasks", "Console Actions", "Task List", "Books", "Help", "Ensemble"}

On success, a custom group managed object is created reflecting the Request Body contents and HTTP status code 201 (Created) is returned.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface action/task permission to the **Grouping** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned and the response body is provided as described in "Response body contents" on page 1002. In addition, the **Location** response header contains the URI of the newly created object.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	261	One of the following errors was detected: • The pattern string specified in **match-info** is not valid. This must be a non-empty string which is a valid regular expression. • One or more of the types specified in **match-info** is invalid. At least one type must be specified, and all must be values as documented for the **match-info types** property.
	290	The requested name is either reserved or already in use.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/groups HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
content-type: application/json
content-length: 74
{
   "description": "Spacely Sprockets Web Servers",
   "name": "SS-Web-Servers"
}
```

Figure 555. Create Custom Group: Request

```
201 Created
server: zSeries management console API web server / 1.0
location: /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:44 GMT
content-type: application/json;charset=UTF-8
content-length: 65
{
   "object-uri": "/api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341"
}
```

Figure 556. Create Custom Group: Response

Delete Custom Group

Use the `Delete Custom Group` operation to delete a custom group. This operation is supported using the BCPii interface.

HTTP method and URI

```
DELETE /api/groups/{group-id}
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Description

If successful, the custom group managed object designated by *{group-id}* is deleted.

If *{group-id}* does not identify an existing custom group, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the custom group designated by *{group-id}*
  - Action/task permission for the **Grouping** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The URI's *{group-id}* does not designate an existing custom Group object, or the API user does not have object-access to the group.

HTTP error status code	Reason code	Description
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
DELETE /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341 HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
```

Figure 557. Delete Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:45 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 558. Delete Custom Group: Response

Add Member to Custom Group

Use the `Add Member to Custom Group` operation to add a member to a custom group. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/groups/{group-id}/operations/add-member
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`object-uri`	String	Required	The object URI of the object to be added to the group

Description

If successful, the managed object designated in the request body attains membership in the custom group identified by *{group-id}*.

The operation is subject to the following restrictions:

- The designated managed object must exist and must not already be a member of the group identified by *{group-id}*
- The group identified by *{group-id}* must be a custom group defined without a pattern-matching specification.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the custom Group object designated by *{group-id}*
  - Object-access permission to the object designated by the request body
  - Action/task permission for the **Grouping** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object and the object designated by the request body.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	291	The designated managed object is already a member of the custom group identified by *{group-id}*.
	294	The group identified by *{group-id}* was defined using pattern-matching.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
	293	The addition of the member to the custom group designated by the URI (*{group-id}*) would introduce a circular reference, which is not permitted.
404 (Not Found)	1	The URI's *{group-id}* does not designate an existing custom Group object, or the API user does not have object-access to the group.
	2	The request body does not designate an existing managed object, or the API user does not have sufficient access to the managed object.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/operations/add-member HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
content-type: application/json
content-length: 78
{
   "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
}
```

Figure 559. Add Member to Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:44 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 560. Add Member to Custom Group: Response

Remove Member from Custom Group

Use the `Remove Member from Custom Group` operation to remove a member from a custom group. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/groups/{group-id}/operations/remove-member
```

In this request, the URI variable *{group-id}* is the object ID of the group

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`object-uri`	String	Required	The object URI of the object to be removed from the group

Description

The managed object designated in the request body relinquishes its membership in the custom group identified by *{group-id}*.

The operation is subject to the following restrictions:

- The managed object designated in the request body must currently be a member of the group identified by *{group-id}*.
- The group identified by *{group-id}* must be a custom group defined without a pattern-matching specification

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the custom Group object designated by *{group-id}*
  - Object-access permission to the object designated by the request body
  - Action/task permission for the **Grouping** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object and the object designated by the request body.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	291	The designated managed object is not a member of the custom group identified by *{group-id}*.
	294	The group identified by *{group-id}* was defined using pattern-matching.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The URI's *{group-id}* does not designate an existing custom Group object, or the API user does not have object-access to the group.
	2	The request body does not designate an existing managed object, or the API user does not have sufficient access to the managed object.
500 (Server Error)	273	An unexpected error occurred during the operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/operations/remove-member HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
content-type: application/json
content-length: 78
{
   "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
}
```

Figure 561. Remove Member from Custom Group: Request

```
204 No Content
date: Fri, 25 Nov 2011 16:45:45 GMT
server: zSeries management console API web server / 1.0
cache-control: no-cache

<No response body>
```

Figure 562. Remove Member from Custom Group: Response

List Custom Group Members

Use the `List Custom Group Members` operation to list custom group members. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/groups/{group-id}/members
```

In this request, the URI variable *{group-id}* is the object ID of the group.

Response body contents

Field name	Type	Description
`members`	Array of nested objects	Array of nested objects which identify members of the group designated by *{group-id}*.

Each nested member object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The value of the member object's **object-uri** property.
`name`	String	The value of the member object's **name** property.

Description

This operation lists the members of the Group object designated by *{group-id}*. The results of this operation only include references to member objects for which the API user has object-access authority.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section. If the group currently has no members, HTTP status code 200 (OK) is returned, along with an empty response body.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the custom Group object designated by *{group-id}*
  - Object-access permission to each member object to be included in the result.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object and each member object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1009.

On error, appropriate HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	1	The group designated by the URI (*{group-id}*) does not exist, or the API user does not have sufficient access (as described above).

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/groups/ee2782af-dd98-3ec0-bc2d-cfe2e9154341/members HTTP/1.1
x-api-session: 42r6t4chltipvd6l4l61wi3111tf7fv2hes80hjqjs3invt7cp
```

Figure 563. List Custom Group Members: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:45:45 GMT
content-type: application/json;charset=UTF-8
content-length: 213
{
   "members": [
      {
         "name": "SS-Web-Svr-1",
         "object-uri": "/api/logical-partitions/576569dc-0db7-11e1-b1f1-f0def14b63af"
      },
      {
         "name": "SS-Web-Svr-2",
         "object-uri": "/api/logical-partitions/588d8c18-0db7-11e1-b1f1-f0def14b63af"
      }
   ]
}
```

Figure 564. List Custom Group Members: Response

Inventory service data

Information about custom groups can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Group objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of the class **"group"** are to be included. An entry for a particular group is included only if the API user has access permission to that object as described in the `Get Custom Group Properties` operation.

For each Group object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Custom Group Properties" on page 1000. That is, the data provided is the same as would be provided if a `Get Custom Group Properties` operation were requested targeting this object.

CPC object

A CPC object represents a managed single Central Processor Complex (CPC).

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This object includes the properties defined in the "Base managed object properties schema" on page 100, with the following class-specific specialization:

Table 489. CPC object: base managed object properties specializations			
Name	**Qualifier**	**Type**	**Description of specialization**
object-uri	—	String/ URI	The canonical URI path of the CPC object, of the form `/api/cpcs/{cpc-id}` where *{cpc-id}* is the value of the **object-id** property of the CPC object.
parent	(pc)	String/ URI	A CPC object has no parent, so this property is always a null object.
class	—	String	The class of a CPC object is **"cpc"**.
name	(ro)(pc)	String (1-8)	The CPC name
description	(ro) if **dpm-enabled** is **false** or (w)(pc) if **dpm-enabled** is **true**	String (0-1024)	The descriptive text associated with this CPC object. This property is a writable, user-supplied value only when this CPC is enabled for DPM.

Table 489. CPC object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
status	(sc)	String Enum	The current operational status of the CPC object. For HMC (available via WSAPI), one of: • **"active"** - all processors on the CPC are operating. This status is applicable only when **dpm-enabled** is **true**. • **"operating"** - all processors on the CPC are operating.. This status is applicable only when **dpm-enabled** is **false**. • **"not-operating"** - the CPC is not IML complete, or all processors are not operating. • **"not-communicating"** - the CPC is not communicating with the HMC. With the exceptions of **object-uri**, **parent**, **class**, **name**, and **status**, the values of CPC properties are unpredictable unless stated otherwise in this data model. • **"exceptions"** - one or more processors on the CPC object are operating and one or more are not operating. • **"status-check"** - the SE is not communicating with the CPC. • **"service"** - the CPC has been placed in service mode. • **"no-power"** - the CPC has no power. • **"service-required"** - the CPC is operating on the last redundant part of a particular type. • **"degraded"** - one or more of the CPC elements are degraded. For SE (available via BCPii), one of: • **"active"** - the CPC is active. This status is applicable only when **dpm-enabled** is **true**. • **"operating"** - the CPC is operational. This status is applicable only when **dpm-enabled** is **false**. • **"not-operating"** - the CPC is not IML complete, or all processors are not operating. • **"exceptions"** - one or more processors on the CPC object are operating and one or more are not operating. • **"status-check"** - the SE is not communicating with the CPC. • **"no-power"** - the CPC has no power. • **"check-stop"** - the CPC is in check stop state.

Table 489. CPC object: base managed object properties specializations (continued)			
Name	**Qualifier**	**Type**	**Description of specialization**
acceptable-status	(w)(pc)	Array of String Enum	The set of operational **status** values in which the CPC object can exist and be considered in an acceptable (not alert causing) state. For HMC (available via WSAPI), zero or more of: • **"active"** - all processors on the CPC are operating. This status is applicable only when **dpm-enabled** is **true**. • **"operating"** - all processors on the CPC are operating.. This status is applicable only when **dpm-enabled** is **false**. • **"not-operating"** - the CPC is not IML complete, or all processors are not operating. • **"not-communicating"** - the CPC is not communicating with the HMC. With the exceptions of **object-uri**, **parent**, **class**, **name**, and **status**, the values of CPC properties are unpredictable unless stated otherwise in this data model. • **"exceptions"** - one or more processors on the CPC object are operating and one or more are not operating. • **"status-check"** - the SE is not communicating with the CPC. • **"service"** - the CPC has been placed in service mode. • **"no-power"** - the CPC has no power. • **"service-required"** - the CPC is operating on the last redundant part of a particular type. • **"degraded"** - one or more of the CPC elements are degraded. For SE (available via BCPii), zero or more of: • **"active"** - all processors on the CPC are operating. This status is applicable only when **dpm-enabled** is **true**. • **"operating"** - all processors on the CPC are operating. This status is applicable only when **dpm-enabled** is **false**. • **"not-operating"** - the CPC is not IML complete, or all processors are not operating. • **"exceptions"** - one or more processors on the CPC object are operating and one or more are not operating. • **"status-check"** - the SE is not communicating with the CPC. • **"no-power"** - the CPC has no power. • **"service-required"** - the CPC is operating on the last redundant part of a particular type. • **"degraded"** - one or more of the CPC elements are degraded. • **"acceptable"** - Indicates all channels are not operating, but their statuses are acceptable.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties.

Table 490. CPC object: class specific additional properties			
Name	**Qualifier**	**Type**	**Description**
se-version	(pc)	String (1-8)	The current release level of the primary SE internal code. For example, "2.11.1". Note that the alternate SE is normally at the same level, except when installing new internal code levels.
has-hardware-messages	(pc)	Boolean	The CPC object has hardware messages (true), or does not have hardware messages (false).
iml-mode	(pc)	String Enum	The Initial Microcode Load (IML) mode type of the CPC object. One of: • **"not-set"** - the CPC is not IMLed or the CPC is not communicating with the HMC. • **"esa390"** - the CPC is in ESA/390 mode • **"lpar"** - the CPC is in logical partition mode • **"esa390-tpf"** - the CPC is in ESA/390 TPF mode • **"dpm"** - the CPC is in DPM mode
dpm-enabled	(pc)	Boolean	The CPC is enabled for DPM (true) or not (false).
auto-start-list	(pc)	Array of objects	The array of nested auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. This is an ordered array. The partitions or partition groups that are earlier in the array are started before those that are later in the array. The order in which the individual partitions within a given partition group are started is not specified. An empty array indicates that no partitions are automatically started. The auto-start list is specified through the `Set Auto-Start List` operation. This property is only present when **dpm-enabled** is **true**.
is-cpacf-enabled	—	Boolean	Whether Central Processor Assist for Cryptographic Functions (CPACF) is enabled (true) for the CPC or not (false).
next-activation-profile-name	(w)(pc)	String (1-16)	For a CPC that is not enabled for DPM, the name of the activation profile to be used on the next activation of the CPC. The **group-uri** query parameter can be used on a `Get CPC Properties` operation to specify the object URI of the Custom Group object used for determining the next activation profile name to be used. If not specified, the system-defined CPC group is used for this determination. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM. **Note:** This property is not permitted on an `Update CPC Properties` operation when the CPC is enabled for DPM.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
last-used-activation-profile	(pc)	String (0-16)	For a CPC that is not enabled for DPM, the name of the activation profile used on the last activation of the CPC or an empty string. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM.
last-used-iocds	(pc)	String (0-3)	For a CPC that is not enabled for DPM, the name of the IOCDS most recently used by the CPC or an empty string if the CPC has not been IMLed. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM.
machine-model	(pc)	String (1-3)	The model of the machine containing this CPC. For example, "M15".
machine-type	(pc)	String (1-4)	The type of the machine containing this CPC. For example, "2817".
machine-serial-number	(pc)	String (1-12)	The serial number of the machine containing this CPC. For example, "00 - SP1D92B".
cpc-serial-number	—	String (1-12)	The serial number of the CPC. For example, "00000SP1D92B".
cpc-node-descriptor	(pc)	String (2)	The hexadecimal number mapped to the device location of the CPC. This property identifies the CPC's relative order among other CPCs, if any, in the machine. For example, "00".
is-cbu-installed	—	Boolean	The Capacity Backup Upgrade (CBU) facility is installed (true), or not installed (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-cbu-enabled	—	Boolean	CBU is enabled (true), or is not enabled (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-cbu-activated	—	Boolean	CBU is activated (true), or is not activated (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-real-cbu-available	—	Boolean	Real CBU is available (true), or not available (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
cbu-activation-date	—	Timestamp	The date of CBU activation. Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
cbu-expiration-date	—	Timestamp	The date of CBU expiration. Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
cbu-number-of-tests-left	—	Integer	The number of CBU tests left. Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-secure-execution-enabled[1]	(pc)	Boolean	If **true**, Secure Execution for Linux is enabled. If **false**, Secure Execution for Linux is disabled.
is-global-key-installed[1]	(pc)	Boolean	If **true**, the Global key is installed. If **false**, the Global key is not installed. If Secure Execution for Linux is not enabled, this property will be returned as a null.
is-host-key-installed[1]	(pc)	Boolean	If **true**, the Host key is installed. If **false**, the Host key is not installed. If Secure Execution for Linux is not enabled, this property will be returned as a null.
global-primary-key-hash[1]	(pc)	String (64)	The lowercase hexadecimal global primary key hash. If a global primary key is not installed, this property will be returned as a null. **Note:** This property is not returned on a `Get CPC Properties` operation when the API user does not have action/task permission to the **Manage Secure Execution Keys** task.
global-secondary-key-hash[1]	(pc)	String (64)	The lowercase hexadecimal global secondary key hash. If a global secondary key is not installed, this property will be returned as a null. **Note:** This property is not returned on a `Get CPC Properties` operation when the API user does not have action/task permission to the **Manage Secure Execution Keys** task.
host-primary-key-hash[1]	(pc)	String (64)	The lowercase hexadecimal host primary key hash. If a host primary key is not installed, this property will be returned as a null. **Note:** This property is not returned on a `Get CPC Properties` operation when the API user does not have action/task permission to the **Manage Secure Execution Keys** task.
host-secondary-key-hash[1]	(pc)	String (64)	The lowercase hexadecimal host secondary key hash. If a host secondary key is not installed, this property will be returned as a null. **Note:** This property is not returned on a `Get CPC Properties` operation when the API user does not have action/task permission to the **Manage Secure Execution Keys** task.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
is-host-import-key-installed	(pc)	Boolean	If **true**, Host Import Key is installed. If **false**, Host Import Key is not installed. If Secure Execution for Linux is not enabled, this property is returned as a null. [Added by feature **secure-execution-key-management**]
primary-host-import-key-id-pattern	(pc)	String (64)	The lowercase hexadecimal primary host import key identification pattern. If a primary host import key is not installed, this property will be returned as a null. **Note:** This property is not returned on a `Get CPC Properties` operation when the API user does not have the task permission to the **Manage Secure Execution Keys** task. [Added by feature **secure-execution-key-management**]
is-service-required	—	Boolean	Whether the CPC is operating using the last redundant part of a particular type (true) or not (false). If true, repairs should be made before additional parts fail that would make this CPC non-operational. Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.
degraded-status	(pc)	Array of String Enum	The set of degraded status values. If the CPC is not degraded, this property contains "not-degraded" as the only value. Otherwise, this property contains one or more of the following: • **"memory"** - loss of memory • **"io"** - loss of I/O channels • **"node"** - one or more books are no longer functioning • **"ring"** - the ring connecting the book is open • **"cbu"** - CBU resources have expired • **"mru"** - cycle time reduction due to an MRU problem • **"ambient-temp"** - cycle time reduction due to a temperature problem • **"iml"** - CPC was IMLed during cycle time reduction.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
processor-running-time-type	(w)	String Enum	For a CPC that is not enabled for DPM, denotes how the processor-running-time property value was determined. One of: • **"system-determined"** - the processor running time is dynamically determined by the system • **"user-determined"** - the processor running time is set to a constant value. **Note:** If **iml-mode** is not **"lpar"**, a null object is returned. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM. **Note:** This property is not permitted on an `Update CPC Properties` operation when the CPC is enabled for DPM.
processor-running-time	(w)	Integer	For a CPC that is not enabled for DPM, the amount of continuous time, in milliseconds, allowed for logical processors to perform jobs on shared processors for the CPC object. Note: a null object is returned if the **iml-mode** is not **"lpar"** or processor-running-time-type is **"system-determined"**. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM. **Note:** This property is not permitted on an `Update CPC Properties` operation when the CPC is enabled for DPM.
does-wait-state-end-time-slice	(w) or — if **se-version** is **"2.14.0"** or later	Boolean	For a CPC that is not enabled for DPM, Logical Partitions of the CPC should lose their share of running time when they enter a wait state (true), or should not lose their share of running time when they enter a wait state (false). Note: a null object is returned if the **iml-mode** is not **"lpar"** or processor-running-time-type is **"system-determined"**. **Note:** This property is not returned on a `Get CPC Properties` operation when the CPC is enabled for DPM. **Note:** This property is not permitted on an `Update CPC Properties` operation when the CPC is enabled for DPM. **Note:** When **se-version** is **"2.14.0"** or later, this property is not permitted on an `Update CPC Properties` operation, and its value is always **false**.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
is-on-off-cod-installed	—	Boolean	On/Off Capacity on Demand is installed for the CPC object (true), or is not installed (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-on-off-cod-enabled	—	Boolean	On/Off CoD is enabled (true), or is not enabled (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
is-on-off-cod-activated	—	Boolean	On/Off CoD is currently activated for the CPC object (true), or is not currently activated (false). Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
on-off-cod-activation-date	—	Timestamp	The date when On/Off CoD was activated. Note: if **status** is **"not-communicating"**, a null object is returned. Refer to the *Capacity On Demand User's Guide* for details.
software-model-permanent	(pc)	String (1-3)	The software model based on the permanent processors and the capacity level that the processors are running on. For example, "723".
software-model-permanent-plus-billable	(pc)	String (1-3)	The software model based on the permanent plus billable processors and the capacity level that the processors are running on. For example, "723".
software-model-permanent-plus-temporary	(pc)	String (1-3)	The software model based on the permanent plus all temporary processors and the capacity level that the processors are running on. For example, "723".
software-model-purchased[4]	—	String (1-3)	The software model based on the purchased processors and the capacity level that the processors are running on. For example, "723".
msu-permanent	—	Integer	The MSU value associated with the software model based on the permanent processors.
msu-permanent-plus-billable	—	Integer	The MSU value associated with the software model based on the permanent plus billable processors.
msu-permanent-plus-temporary	—	Integer	The MSU value associated with the software model based on the permanent plus all temporary processors.
msu-purchased[4]	—	Integer	The MSU value associated with the software model based on the purchased processors.
processor-count-general-purpose	(pc)	Integer	The count of active general purpose processors.
processor-count-service-assist	—	Integer	The count of active service assist processors (SAP).
processor-count-aap	—	Integer	The count of active IBM zEnterprise Application Assist Processor (zAAP) processors.
processor-count-ifl	(pc)	Integer	The count of active Integrated Facility for Linux (IFL) processors.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
processor-count-icf	(pc)	Integer	The count of active Internal Coupling Facility (ICF) processors.
processor-count-iip	(pc)	Integer	The count of active IBM z Integrated Information Processor (zIIP) processors.
processor-count-defective	—	Integer	The count of defective processors. Includes all processor types.
processor-count-spare	—	Integer	The count of spare processors. Includes all processor types.
processor-count-pending	—	Integer	The combined number of processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records.
processor-count-pending-general-purpose	—	Integer	The number of general purpose processors that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-pending-service-assist[3]	—	Integer	The number of service assist processors (SAP) that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-pending-aap[3]	—	Integer	The number of Application Assist processors (zAAP) that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-pending-ifl[3]	—	Integer	The number of Integrated Facility for Linux processors (IFL) that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-pending-icf[3]	—	Integer	The number of Integrated Coupling Facility processors (ICF) that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-pending-iip[3]	—	Integer	The number of z Integrated Information Processors (zIIP) that will become active, when more processors are made available by adding new hardware or by deactivating capacity records. Note: if **status** is **"not-communicating"**, a null object is returned.
processor-count-permanent-service-assist[4]	—	Integer	The count of permanent service assist processors (SAP).
processor-count-permanent-ifl[4]	—	Integer	The count of permanent Integrated Facility for Linux (IFL) processors.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
processor-count-permanent-icf[4]	—	Integer	The count of permanent Internal Coupling Facility (ICF) processors.
processor-count-permanent-iip[4]	—	Integer	The count of permanent z Integrated Information Processor (zIIP) processors.
processor-count-unassigned-service-assist[4]	—	Integer	The count of unassigned service assist processors (SAP).
processor-count-unassigned-ifl[4]	—	Integer	The count of unassigned Integrated Facility for Linux (IFL) processors.
processor-count-unassigned-icf[4]	—	Integer	The count of unassigned Internal Coupling Facility (ICF) processors.
processor-count-unassigned-iip[4]	—	Integer	The count of unassigned z Integrated Information Processor (zIIP) processors.
has-temporary-capacity-change-allowed	—	Boolean	Whether API applications are allowed to make changes to temporary capacity (true), or not (false).
ec-mcl-description	—	ec-mcl-description object	Describes the Engineering Change (EC) and MicroCode Level (MCL) for the CPC object. An empty object is returned if the information is unavailable from the SE. Refer to the description of the `ec-mcl-description` object for details.
has-automatic-se-switch-enabled	—	Boolean	Automatic switching between primary and alternate Support Elements is enabled for the CPC object (true), or is not enabled (false). Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.
stp-configuration	—	stp-config object	Describes the Server Time Protocol (STP) configuration. Refer to the description of the `stp-config` object for details. Note: if the required feature(s) are not installed, the property is not returned.
lan-interface1-type	(pc)	String Enum	The adapter type of the Support Element's LAN interface 1. One of the following: • **"ethernet"** • **"token-ring"** • **"unknown"**
lan-interface1-address	(pc)	String (1-12)	The MAC address of the Support Element's LAN interface 1.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
lan-interface2-type	(pc)	String Enum	The adapter type of the Support Element's LAN interface 2. One of the following: • **"ethernet"** • **"token-ring"** • **"unknown"**
lan-interface2-address	(pc)	String (1-12)	The MAC address of the Support Element's LAN interface 2.
network1-ipv4-mask	(pc)	String (1-15)	The network IP mask value.
network1-ipv4-pri-ipaddr	(pc)	String IPV4 address	The primary IPv4 address or a null object if not configured.
network1-ipv4-alt-ipaddr	(pc)	String IPV4 address	The alternate IPv4 address or a null object if not configured.
network1-ipv6-info	—	Array of ipv6-info objects	A list of objects describing the Support Element's IPv6 network connections. If no IPv6 connections are defined, an empty list is returned.
network2-ipv4-mask	(pc)	String (1-15)	The network IP mask value.
network2-ipv4-pri-ipaddr	(pc)	String IPV4 address	The primary IPv4 address or a null object if not configured.
network2-ipv4-alt-ipaddr	(pc)	String IPV4 address	The alternate IPv4 address or a null object if not configured.
network2-ipv6-info	—	Array of ipv6-info objects	A list of objects describing the Support Element's IPv6 network connections. If no IPv6 connections are defined, an empty list is returned.
hardware-messages	(c)(pc)	Array of hardware-message objects	The complete list of all CPC hardware messages, each identified by its URI. This list corresponds to the list provided by the `List CPC Hardware Messages` operation. If the CPC has no hardware messages, then an empty array is provided. The list of returned hardware messages can change as a result of new messages being dynamically added or removed by the infrastructure or due to hardware messages being deleted through the `Delete CPC Hardware Message` operation. **Note:** This property is not returned by the `Get CPC Properties` operation, and only sessions associated with an HMC user with permission to the Hardware Messages task will receive a property-change notification for this property.
storage-total-installed	(pc)	Long	Amount of installed storage, in megabytes.
storage-hardware-system-area	—	Long	Amount of storage, in megabytes, reserved for the base hardware system area (HSA).

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
storage-customer	—	Long	Amount of storage, in megabytes, for use by the customer.
storage-customer-central	—	Long	Amount of storage, in megabytes, which is the central storage in use across the active partitions.
storage-customer-expanded	—	Long	Amount of storage, in megabytes, which is the expanded storage in use across the active partitions. When the **iml-mode** is **"dpm"** this property will be set to 0.
storage-customer-available	—	Long	Amount of storage, in megabytes, which is not in use.
storage-vfm-increment-size[1]	—	Long	The increment size of any IBM Virtual Flash Memory (VFM) storage property, in gigabytes (GB), on this CPC and its logical partitions or partitions.
storage-vfm-total[1]	—	Long	The total amount of VFM storage, in gigabytes (GB), installed on this CPC. The valid value should be a multiple of the value indicated on the **storage-vfm-increment-size** property for this CPC.
maximum-hipersockets	—	Integer	The maximum number of HiperSocket adapters that may be created for the CPC when the CPC is enabled for DPM. This property will be omitted if **dpm-enabled** is **false**.
maximum-alternate-storage-sites	—	Integer	The maximum number of alternate storage-site instances that may be added to the FICON configuration associated with this CPC. This property will be omitted if **dpm-enabled** is **false**.
available-features-list	—	Array of cpc-feature-info objects	The list of optional features or behavior supported by this CPC. If the CPC has no optional features, then an empty array is provided.
maximum-partitions	—	Integer	The maximum number of partitions that this CPC supports. This property will be omitted if **dpm-enabled** is **false**.
management-world-wide-port-name	(pc)	String (16)	The worldwide port name that the CPC has defined to be used to discover Tape Library objects. This value will be **null** if the Request Tape Library Zoning request has not been issued. This value is only available when the **dpm-fcp-tape-management** feature is enabled on the target CPC.
sna-name	—	String (1-17)	The fully qualified SNA name of the Console.

Table 490. CPC object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
maximum-ism-vchids	—	Integer	The maximum number of Internal Shared Memory (ISM) adapters that may be created for the CPC when the CPC is enabled for DPM. This property will be omitted if **dpm-enabled** is **false**.
minimum-fid-number	—	Integer	The minimum number that can be specified for a Function ID (FID) that may be created for the CPC when the CPC is enabled for DPM. This property will be omitted if **dpm-enabled** is **false**.
maximum-fid-number	—	Integer	The maximum number that can be specified for a Function ID (FID) that may be created for the CPC when the CPC is enabled for DPM. This property will be omitted if **dpm-enabled** is **false**.

Notes:

1. On a Get request, this property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
2. On a Get request, this property is returned only when the SE version is 2.14.0 or later.
3. This property is not supported and is always 0 when the SE version is 2.16.0 or later.
4. This property is returned only when the SE version is 2.16.0 or later.

Table 491. ipv6-info object properties		
Name	**Type**	**Description**
type	String Enum	The IPv6 scope. One of the following values: • **"link-local"** • **"static"** • **"auto"**
prefix	Integer	The number of leading bits of the IPv6 address that represent the network prefix
pri-ip-address	String IPV6 address	The primary IPv6 address
alt-ip-address	String IPV6 address	The alternate IPv6 address or a null object if not configured

Table 492. hardware-message object properties		
Name	**Type**	**Description**
element-uri	String/ URI	The canonical URI path of the CPC hardware message. The URI is in the following form: `/api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}`, where *{hardware-message-id}* is the value of the **element-id** property of the hardware message.
element-id	String (36)	The unique identifier for the hardware message. The **element-id** is in the form of a UUID.
parent	String/ URI	The parent of a CPC hardware message is the CPC object. The **parent** value is the canonical URI path for the CPC.
class	String	The **class** of a hardware message object is **"hardware-message"**.
timestamp	Timestamp	The **timestamp** represents the date and time when the hardware message was created.
service-supported	Boolean	Indicates whether or not this hardware message represents a problem for which service may be requested. **True** is returned if this hardware message supports service, or **false** is returned if it does not.
text	String	The text of the hardware message.
details	Object	A hardware-message-details-object if there are hardware message details, or null if there are no hardware message details.

Table 493. cpc-feature-info object properties		
Name	**Type**	**Description**
name	String Enum	The name of the feature. One of: • **"dpm-storage-management"** - Indicates that the CPC supports Storage Groups and FICON storage resources. FCP and FICON storage resources are defined in Storage Groups, which are then attached to this CPC's partitions. If the CPC does not have this feature, FCP storage resources are represented by HBAs, which must be directly attached to this CPC's Partitions. This feature is applicable only for CPCs enabled for DPM. • **"dpm-fcp-tape-management"** - Indicates that the CPC supports Tape Libraries linked through FCP connections. FCP tape resources are defined in Tape Links, which are then attached to this CPC's partitions. This feature is applicable only for CPCs enabled for DPM. • **"dpm-smcd-partition-link-management"** - Indicates that the CPC supports Partition Links via SMC-D connections. This feature is applicable only for CPCs enabled for DPM. See Chapter 6, “Features,” on page 103 for a list of operations that are affected for each of these features.
description	String	A brief description of the feature.

Table 493. cpc-feature-info object properties (continued)		
Name	**Type**	**Description**
state	Boolean	Indicates if the feature is currently enabled (**true**) or disabled (**false**) for this CPC.

auto-start-entry object

An auto-start-entry object specifies either a partition or a group of partitions that are automatically started when the CPC is started. Every auto-start-entry object contains the following base properties in addition to the type-specific properties:

Table 494. auto-start-entry object base properties		
Name	**Type**	**Description**
post-start-delay	Integer	Amount of time, in seconds, to wait before starting the next partition or group of partitions. Must be greater than or equal to 0.
type	String Enum	The type of auto-start-entry. One of • **"partition"** - this entry identifies a single partition. • **"partition-group"** - this entry identifies a partition group.

An auto-start-entry object with a **type** value of **"partition"** identifies a single partition. In addition to the base properties, it contains the following type-specific properties:

*Table 495. auto-start-entry object type-specific properties when **type** value is **"partition"***		
Name	**Type**	**Description**
partition-uri	String/ URI	The URI of the partition to start.

An auto-start-entry object with a **type** value of **"partition-group"** identifies a group of partitions. In addition to the base properties, it contains the following type-specific properties:

*Table 496. auto-start-entry object type-specific properties when **type** value is **"partition-group"***		
Name	**Type**	**Description**
name	String (1-64)	The name of this partition group. This name must be unique among all partition groups of this CPC. The length and character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100.
description	String (0-1024)	The Description for this partition group.
partition-uris	Array of String/ URI	The list of one or more partitions to start, represented by their URIs. This list is unordered. The partitions in a group can be started in any order or simultaneously by the system.

Energy management related additional properties

In addition to the properties defined in "Class specific additional properties" on page 1013, this object includes the following additional class-specific properties related to energy management. For further explanation of the various states involved, please see "Special states" on page 1385.

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

Table 497. CPC object: energy management related additional properties

Name	Qualifier	Type	Description
cpc-power-rating	—	Integer	Specifies the maximum power draw in watts (W) of this CPC. This is a calculated value as indicated by the electrical rating labels or system rating plates of the CPC components.
cpc-power-consumption	(mg)	Integer	Specifies the current power consumption in watts (W) for this CPC. The CPC power consumption includes the power consumption of the zCPC and hardware extensions. If the system does not include any hardware extensions, the CPC power consumption will be equal to the zCPC power consumption.
cpc-power-saving	—	String Enum	Specifies the current power saving setting of the CPC. Power saving is used to reduce the energy consumption of a system and can be managed in the Set Power Saving operation. The possible settings include: • **"high-performance"** - The power consumption and performance of the CPC are not reduced. This is the default setting. • **"low-power"** - All components of the CPC enabled for power saving will have reduced performance to allow for low power consumption. • **"custom"** - Custom mode indicates that some, but not all, components of the CPC are in the Low power setting. • **"not-supported"** - Power saving is not supported for this CPC. • **"not-available"** - Specifies that **cpc-power-saving** property could not be read from this CPC. • **"not-entitled"** - The server is not entitled for Power saving.
cpc-power-saving-state	—	String Enum	Specifies the power saving setting of the CPC set by the user. Please note that this property indicates the user setting and may not match the real state of the hardware compared to the **cpc-power-saving** property. For more information, see "Group power saving" on page 1386. The possible settings include: • **"high-performance"** - Specifies not reducing the power consumption and performance of the CPC. • **"low-power"** - Specifies low power consumption for all components of the CPC enabled for power saving. • **"custom"** - Specifies that the CPC does not control the children. This is the default setting. • **"not-supported"** - Specifies that power saving is not supported for this CPC. • **"not-entitled"** - Specifies that the server is not entitled to power saving.

Table 497. CPC object: energy management related additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
cpc-power-save-allowed	—	String Enum	Should be used to determine if a call of the power save operation is currently allowed. If a value other than **"allowed"** is returned the caller may reckon that the power save operation will fail. The possible settings include: • **"allowed"** - Alter power save setting is allowed for this CPC • **"unknown"** - Unknown reason • **"not-supported"** - Power saving is not supported for this CPC. • **"not-entitled"** - Specifies the server is not entitled to power capping.
cpc-power-capping-state	—	String Enum	Specifies the current power capping setting of the CPC. Power capping is used to limit peak power consumption of a system and can be managed in the Set Power Cap operation. The possible settings include: • **"disabled"** - The power cap of the CPC is not set and the peak power consumption is not limited. This is the default setting. • **"enabled"** - All components of the CPC available for power capping will be capped to limit the peak power consumption of the CPC. • **"custom"** - The components of the CPC can be individually configured for power capping. • **"not-supported"** - Power capping is not supported for this CPC. • **"not-entitled"** - The server is not entitled for Power capping.
cpc-power-cap-minimum	—	Integer	Specifies the minimum value for the CPC cap value in watts (W). This is a sum of the component minimum cap values. If the **cpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383
cpc-power-cap-maximum	—	Integer	Specifies the maximum value for the CPC cap value in watts (W). This is a sum of the component maximum cap values. If the **cpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383
cpc-power-cap-current	—	Integer	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the CPC and is the sum of the component Cap values. If the **cpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383

Table 497. CPC object: energy management related additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
cpc-power-cap-allowed	—	String Enum	Should be used to determine if a call of the power capping operation is currently allowed. If a value other than **"allowed"** is returned the caller may reckon that the power capping operation will fail. The possible settings include: • **"allowed"**- Alter power capping setting is allowed for this CPC • **"unknown"** - Unknown reason • **"not-supported"** - Power capping is not supported for this CPC. • **"not-entitled"** - Specifies the server is not entitled to power capping.
zcpc-power-rating	—	Integer	Specifies the maximum power draw in watts (W) of this zCPC. This is a calculated value as indicated by the electrical rating labels or system rating plates of the zCPC components.
zcpc-power-consumption	(mg)	Integer	Specifies the current power consumption of the zCPC in watts (W).
zcpc-power-saving	—	String Enum	Specifies the current power saving setting of the zCPC. Power saving is used to reduce the energy consumption of a system and can be managed in the Set Power Saving operation. The possible settings include: • **"high-performance"** - The power consumption and performance of the zCPC are not reduced. This is the default setting. • **"low-power"** - The performance of the zCPC is reduced to allow for low power consumption. • **"not-supported"** - Power saving is not supported for this zCPC. • **"not-available"** - Specifies that **zcpc-power-saving** property could not be read for this zCPC. • **"not-entitled"** - The server is not entitled for Power saving.

Table 497. CPC object: energy management related additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
zcpc-power-saving-state	—	String Enum	Specifies the power saving setting of the zCPC set by the user. Please note that this property indicates the user setting and may not match the real state of the hardware compared to the **zcpc-power-saving** property. For more information, see "Group power saving" on page 1386. The possible settings include: • **"high-performance"** - Specifies not reducing the power consumption and performance of the zCPC. This is the default setting. • **"low-power"** - Specifies low power consumption for all components of the zCPC enabled for power saving. • **"not-supported"** - Specifies that power saving is not supported for this zCPC. • **"not-entitled"** - Specifies that the server is not entitled to power saving.
zcpc-power-save-allowed	—	String Enum	Should be used to determine if a call of the power save operation is currently allowed. If a value other than **"allowed"** is returned the caller may reckon that the power save operation will fail. The possible settings include: • **"allowed"** - Alter power save is allowed for this zCPC • **"unknown"** - Unknown reason • **"not-entitled"** - Specifies the server is not entitled to power save. • **"under-group-control"** - The zCPC is under group control and cannot be individually altered. • **"not-supported"** - Power saving is not supported for this zCPC. • **"once-a-day-exceeded"** - Power saving mode has been entered at some point during the day and will not be allowed again until the next calendar day.
zcpc-power-capping-state		String Enum	Specifies the current power capping setting of the zCPC. Power capping is used to limit peak power consumption of a system and can be managed in the Set Power Cap operation. The possible settings include: • **"disabled"** - The power cap of the zCPC is not set and the peak power consumption is not limited. This is the default setting. • **"enabled"** - The peak power consumption of the zCPC is limited to the Current cap value. • **"custom"** - The components of the CPC can be individually configured for power capping. • **"not-supported"** - Power capping is not supported for this zCPC. • **"not-entitled"** - The server is not entitled for Power capping.

Table 497. CPC object: energy management related additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
zcpc-power-cap-minimum	—	Integer	Specifies the minimum value for the zCPC cap value in watts (W). If the **zcpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383
zcpc-power-cap-maximum	—	Integer	Specifies the maximum value for the zCPC cap value in watts (W). If the **zcpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383
zcpc-power-cap-current	—	Integer	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the zCPC. If the **zcpc-power-cap-allowed** property value is **"not-entitled"**, the value is **null**. For more information on entitlement, see Chapter 12, “Energy management,” on page 1383
zcpc-power-cap-allowed	—	String Enum	Should be used to determine if a call of the power capping operation is currently allowed. If a value other than **"allowed"** is returned the caller may reckon that the power capping operation will fail. The possible settings include: • **"allowed"** - Alter power capping is allowed for this zCPC • **"unknown"** - Unknown reason • **"not-entitled"** - Specifies the server is not entitled to power cap. • **"not-supported"** - Power capping is not supported for this zCPC. • **"under-group-control"** - Power capping is under group control
zcpc-ambient-temperature	(mg)	Float	Specifies the input air temperature in degrees Celsius (°C) as measured by the system.
zcpc-exhaust-temperature	—	Float	Specifies the exhaust air temperature in degrees Celsius (°C) as calculated by the system. This is useful in determining potential hot spots in the data center.
zcpc-humidity	(mg)	Integer	Specifies the amount of water vapor in the air as measured by the system. The humidity sensor gives a reading of the relative humidity of the air entering the system. The recommended long-term relative humidity for a system with an altitude from sea level to 900 meters (2953 feet) is 60%. The range of acceptable relative humidity is 8% - 80%. For more information, see the chapter related to environmental specifications in the *Installation Manual for Physical Planning*.

Table 497. CPC object: energy management related additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
zcpc-dew-point	(mg)	Float	Specifies the air temperature in degrees Celsius (°C) at which water vapor will condense into water. This is a calculated value based on the current temperature and relative humidity. Cooling the server to the dew point can result in condensation on critical internal parts, leading to equipment failure, unless the computer room environment is adequately maintained to prevent it. For more information, see the chapter related to environmental specifications in the *Installation Manual for Physical Planning*.
zcpc-heat-load	(mg)	Integer	Specifies the amount of heat in Btu/hr. removed from the system.
zcpc-heat-load-forced-air	—	Integer	Specifies the amount of heat in Btu/hr. removed from the system by forced-air.
zcpc-heat-load-water	—	Integer	Specifies the amount of heat in Btu/hr. removed from the system by chilled water. The value is always 0 on an air cooled system.
zcpc-maximum-potential-power	—	Integer	Specifies the maximum potential power consumption of a system in watts (W). This value is based on the configuration of the system and can be used for power and cooling planning.
zcpc-maximum-potential-heat-load	—	Integer	Specifies the maximum potential heat load of a system in Btu/hr. This value is based on the configuration of the system and can be used for power and cooling planning.
last-energy-advice-time	(pc)	Timestamp	The timestamp of the most recent change to the energy optimization advice for the CPC or **null** if the timestamp is not available. A value of **-1** indicates that no advice has been generated for this CPC. This property is only provided when the associated SE is at version 2.13.1 or later.
zcpc-minimum-inlet-air-temperature	—	Integer	Minimum allowable operating value in degrees Celsius (°C).
zcpc-maximum-inlet-air-temperature	—	Integer	Maximum allowable operating value in degrees Celsius (°C).
zcpc-maximum-inlet-liquid-temperature	—	Integer	Maximum allowable operating value in degrees Celsius (°C).
zcpc-environmental-class	—	String	Class according to ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) Environmental Classes for IT equipment.

Remote Firmware Update CPC element object

A Remote Firmware Update element object defines a firmware update operation that is scheduled to occur on the CPC at a future time.

Table 498. CPC object - Remote Firmware Update element object properties			
Name	**Qualifier**	**Type**	**Description**
element-id	—	String (36)	The unique identifier for the remote firmware update instance.
element-uri	—	String/ URI	The canonical URI path for the remote firmware update object, of the form `/api/cpcs/{cpc-id}/remote-firmware-updates/{remote-firmware-update-id}`, where `{cpc-id}` is the **object-id** of the containing CPC, and `{remote-firmware-update-id}` is the **element-id** of this remote firmware update.
parent	—	String/ URI	The parent of a remote firmware update is conceptually its owning CPC, and so the parent value is the canonical URI path for the CPC.
class	—	String (22)	The class of a Remote Firmware Update element is **"remote-firmware-update"**.
creation-time	—	Timestamp	The time at which the remote firmware update was scheduled.
scheduled-execution-time	—	Timestamp	The time at which the remote firmware update will begin.
execution-window	—	Integer	The number of minutes the operation will wait if it is blocked at its scheduled execution time, for example due to a busy condition.
execution-percentage	(pc)	Integer (0-100)	The current percentage of this firmware update operation that has completed. [Added by feature **rcl-progress**]
execution-steps	(pc)	Array of Remote Firmware Update Execution Step CPC objects	Array of objects that describe the steps in this firmware update operation. [Added by feature **rcl-progress**]
target-bundle	—	String	The target bundle level for the firmware update.

Table 498. CPC object - Remote Firmware Update element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
state	(pc)	String Enum	The execution state of the scheduled operation. Values: • **"scheduled"** - The current time is not beyond the **"scheduled-execution-time"**. • **"running"** - The operation is currently executing. • **"scheduled-on-peer"** - The operation is currently in a **"scheduled"** state on the peer. This state is only applicable when the target of the operation is a console defined as an HMA. • **"running-on-peer"** - The operation is currently in a **"running"** state on the peer. This state is only applicable when the target of the operation is a console defined as an HMA. • **"pending"** - The operation is complete pending a condition. In order to compete the operation, an additional action specified within the **"pending-conditions"** property will have to be fulfilled. [Added by feature **rcl-history**] • **"succeeded"** - The operation completed successfully. If the target of the operation is a specified CPC, then the state value also indicates that the operation completed without any pending conditions. [Added by feature **rcl-history**] • **"failed"** - The operation failed. The **execution-steps** list should contain a Remote Firmware Update Execution Step CPC object with a **state** value of **"failed"** to indicate the precise step of failure. [Added by feature **rcl-history**]

Table 498. CPC object - Remote Firmware Update element object properties (continued)			
Name	**Qualifier**	**Type**	**Description**
pending-conditions	(pc)	Array of String Enum	The list of conditions that are pending upon the completion of the remote firmware update. Even if the following condition(s) are completed, the **state** of the operation will still remain as **"pending"**. The value of this property will be an empty array if the state of the operation is not **"pending"**. • **"cryptos-or-channels-currently-pending-config-off-on"** - One or more channels and/or cryptos require a configure off followed by a configure on to apply the new firmware. See the **Manage Adapter Firmware UI** task. • **"coupling-facility-partitions-pending-reactivation"** - One or more coupling facility partitions require an activation to apply the new firmware. See the **Query Coupling Facility Partition Reactivations** task. • **"cpc-changes-pending-power-on-reset"** - The system requires a power-on reset to apply the new firmware. See the **Query Internal Code Changes Pending Power On Reset** task. • **"pci-partitions-pending-update"** - One or more PCI partitions require an update to apply the new firmware. See the **Manage PCI System Services** task. • **"pci-adapters-pending-update"** - One or more PCI adapters require an update to apply the new firmware. See the **Update PCI Adapter Internal Code** task. [Added by feature **rcl-history**]
scheduling-console-name	—	String	Name of the console from which the remote firmware update was scheduled. [Added by feature **rcl-progress**]
service-contact-name	—	String	The name of the service representative that scheduled the operation. The value may be empty.
service-contact-telephone-number	—	String	The telephone number of the service representative that scheduled the operation. The value may be empty.
service-contact-email-address	—	String	The email address of the service representative that scheduled the operation. The value may be empty.

Remote Firmware Update Execution Step CPC nested object

A Remote Firmware Update Execution Step CPC nested object describes a step in the overall Console firmware update process. [Added by feature **rcl-progress**]

Table 499. CPC object - Remote Firmware Update Execution Step CPC nested object properties			
Name	**Qualifier**	**Type**	**Description**
id	—	String Enum	Identifies the step whose progress is described by this object Values • **"verify-environment"** - Checks that change management is enabled and other conditions that would cause the remote code load to fail. • **"back-up-critical-data"** - Makes a backup of the targeted platform so that if the 1U server needs to be replaced, it can be restored from this data. • **"accept-installed-changes"** - Makes the previously activated bundle permanent so that it cannot be backed off. • **"retrieve-internal-code-changes"** - Pulls the latest internal code changes that have been released to zRSF. • **"apply-internal-code-changes"** - Activates the internal code changes up to the bundle requested in the remote code load. • **"mirror-internal-code-changes"** - Copies all of the code and data from the primary SE to the alternate SE so that if there is a catastrophic issue on the primary SE, the alternate SE can take over as primary. • **"transmit-system-availability-data"** - Collects data including system status and send it back to the remote support system.
state	—	String Enum	The execution state of the firmware update step. The progression of states for a particular execution step should be **"not-started"**, then **"running"**, and then finally to either **"succeeded"** or **"failed"**. The execution state for a particular execution step will remain as **"not-started"** if a failure occurred at a previous execution step. Values: • **"not-started"** - The step has not started yet. • **"running"** - The step is currently executing. • **"succeeded"** - The step completed successfully. • **"failed"** - The step failed.

List CPC Objects

The `List CPC Objects` operation returns a list of the zManager Web Services API capable managed CPCs. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs
```

Query Parameters

Name	Type	Rqd/Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`cpcs`	Array of cpc-info objects	Array of nested cpc-info objects (described in the next table). If no matching CPC objects are found, an empty array is returned.

Each nested cpc-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	Canonical URI path of the CPC object
`name`	String	The name of the CPC object
`status`	String Enum	The current status of the CPC object **Note:** This property is only returned when the web services interface was used for the request.
`has-unacceptable-status`	Boolean	The **has-unacceptable-status** property of the CPC object. **Note:** This property is only returned when the web services interface was used for the request.
`dpm-enabled`	Boolean	True if the CPC is enabled for DPM; false otherwise. **Note:** This property is only returned when the web services interface was used for the request.
`se-version`	String	The **se-version** property of the CPC object.
`target-name`	String (1-17)	The **target-name** property of the CPC object. The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
`location`	String Enum	The relationship of the origin of the request to the CPC object. One of the following values: • **"local"** - The CPC object is hosting the partition that originated the request. • **"remote"** - The CPC object is not hosting the partition that originated the request. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

For the web services interface this operation lists the zManager Web Services API capable CPC objects that are managed by this HMC. For the BCPii interface this operation lists zManager Web Services API capable CPC objects that can be reached from the source system. The object URI, object ID and display name are provided for each CPC returned. CPCs that are not zManager Web Services API capable are not returned.

If the **name** query parameter is specified, the returned list is limited to those CPC objects that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

For the web services interface, an object is only included in the list if the API user has object-access permission for that object.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, object-access permission to any CPC object to be included in the result.
- For the BCPii interface any partition with send BCPii security controls permission can issue this request to the hosting CPC of the partition.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1037.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
403 (Forbidden	0	The request used the BCPii interface and targeted a system other than the local system.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs HTTP/1.1
x-api-session: 2jm2h7j25d1e1g5wbygmfriyjiit8tp4iqiw8h09j8kz68i0k6
```

Figure 565. List CPC Objects: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2016 07:18:42 GMT
content-type: application/json;charset=UTF-8
content-length: 725
{
   "cpcs": [
      {
         "dpm-enabled": true,
         "has-unacceptable-status": true,
         "name": "P0LXSMOZ",
         "object-uri": "/api/cpcs/e8753ff5-8ea6-35d9-b047-83c2624ba8da",
         "se-version": "2.13.1"
         "status": "not-operating"
      },
      {
         "dpm-enabled": true,
         "has-unacceptable-status": false,
         "name": "R32",
         "object-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
         "se-version": "2.14.0"
         "status": "operating"
      },
      {
         "dpm-enabled": false,
         "has-unacceptable-status": true,
         "name": "ICHABOD",
         "object-uri": "/api/cpcs/ac15c987-90c6-3526-854e-4c612939260d",
         "se-version": "2.13.1"
         "status": "not-operating"
      }
   ]
}
```

Figure 566. List CPC Objects: Response

Get CPC Properties

The `Get CPC Properties` operation retrieves the properties of a single CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the CPC object's data model.
`cached-acceptable`	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**. The default is **false**.
`group-uri`	String/ URI	Optional	The object URI for the Custom Group object to be used for determining the value of the **next-activation-profile-name** property. If omitted the system-defined CPC group will be used.

Response body contents

On successful completion, the response body provides the current values of the properties for the CPC object as defined in "Data model" on page 1010.

Description

Some CPC properties are only available through the web services interface if the HMC is communicating with the SE, and are returned as null objects if the HMC is not communicating with the SE. With the exceptions of **object-uri**, **parent**, **class**, **name**, and **status**, the values of CPC properties are unpredictable unless stated otherwise in the "Data model" on page 1010. This is not the case when using the BCPii interface because the requests are processed directly on the SE.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some of this object's property values are periodically fetched from the Support Element and cached on the HMC for quick access by the web services APIs. Due to the nature of this caching support, the HMC's cached value of a property may differ from the value on the Support Element at any point in time. While the HMC strives to keep the cache reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned. This is also the case for requests using the BCPii interface.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined "Data model" on page 1010.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*.
  - Action/task permission for the **Manage Secure Execution Keys** task to get **global-primary-key-hash**, **global-secondary-key-hash**, **host-primary-key-hash**, **host-secondary-key-hash**, and **primary-host-import-key-id-pattern** properties. [Added by feature **secure-execution-key-management**]
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1040.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	272	Unable to obtain Server Time Protocol (STP) configuration. Retry the request later.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5 HTTP/1.1
x-api-session: 4tcxomq5kfzlvi48mn05a8wzmjc1rzhdf59tak9zg32jynapvh
```

Figure 567. Get CPC Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 18 Sep 2019 22:11:13 GMT
content-type: application/json;charset=UTF-8
content-length: 5549
{
   "acceptable-status":[
      "operating"
   ],
   "additional-status":"",
   "available-features-list":[
      {
         "description":"The DPM storage management approach in which FCP and FICON storage
resources are defined in Storage Groups, which are attached to Partitions.",
         "name":"dpm-storage-management",
         "state":true
      }
   ],
   "cbu-activation-date":0,
   "cbu-expiration-date":0,
   "cbu-number-of-tests-left":0,
   "class":"cpc",
   "cpc-node-descriptor":"00",
   "cpc-power-cap-allowed":null,
   "cpc-power-cap-current":null,
   "cpc-power-cap-maximum":null,
   "cpc-power-cap-minimum":null,
   "cpc-power-capping-state":null,
   "cpc-power-consumption":742,
   "cpc-power-rating":16628,
   "cpc-power-save-allowed":null,
   "cpc-power-saving":null,
   "cpc-power-saving-state":null,
   "cpc-serial-number":"00000005742D",
   "degraded-status":[
      "not-degraded"
   ],
   "description":"Central Processing Complex (CPC)",
   "does-wait-state-end-time-slice":null,
   "dpm-enabled":false,
   "ec-mcl-description":{
      "action":[
         {
            "activation":"current",
            "pending":false,
            "type":"channel-config"
         },
         {
            "activation":"current",
            "pending":false,
            "type":"coupling-facility-reactivation"
         },
```

Figure 568. Get CPC Properties: Response (Part 1)

```
        {
           "activation":"current",
           "pending":false,
           "type":"power-on-reset-tracking"
        },
        {
           "activation":"next",
           "pending":false,
           "type":"channel-config"
        },
        {
           "activation":"next",
           "pending":false,
           "type":"coupling-facility-reactivation"
        },
        {
           "activation":"next",
           "pending":false,
           "type":"power-on-reset-tracking"
        }
     ],
     "ec":[
        {
           "description":"SE Framework",
           "mcl":[
              {
                 "last-update":null,
                 "level":"000",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46598",
           "part-number":"02WF216",
           "type":"Base EC"
        },
```


Figure 569. Get CPC Properties: Response (Part 2)

```
        {
           "description":"Concurrent Upgrade Sync Point",
           "mcl":[
              {
                 "last-update":null,
                 "level":"000",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46599",
           "part-number":"02WF217",
           "type":"Other Optional EC"
        },
        {
           "description":"SE Licensed Internal Code Alerts",
           "mcl":[
              {
                 "last-update":null,
                 "level":"000",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46600",
           "part-number":"02WF218",
           "type":"Other Optional EC"
        },
```


Figure 570. Get CPC Properties: Response (Part 3)

```
        {
           "description":"Embedded Operating System",
           "mcl":[
              {
                 "last-update":null,
                 "level":"000",
                 "type":"retrieved"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"activated"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"accepted"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"installable-concurrent"
              },
              {
                 "last-update":null,
                 "level":"000",
                 "type":"removable-concurrent"
              }
           ],
           "number":"P46597",
           "part-number":"02WF215",
           "type":"Base EC"
        }
     ]
  },
  "global-primary-key-hash": null,
  "global-secondary-key-hash": null,
  "has-automatic-se-switch-enabled":false,
  "has-hardware-messages":true,
  "has-temporary-capacity-change-allowed":false,
  "has-unacceptable-status":false,
  "host-primary-key-hash": "0dfeabcd012345670dfeabcd012345670dfeabcd012345670dfeabcd01234567",
  "host-secondary-key-hash": "febc789012341234febc789012341234febc789012341234
     febc789012341234",
  "iml-mode":"lpar",
  "is-cbu-activated":false,
  "is-cbu-enabled":false,
  "is-cbu-installed":false,
  "is-cpacf-enabled":false,
  "is-host-import-key-installed":true,
```


Figure 571. Get CPC Properties: Response (Part 4)

```
    "is-locked":false,
    "is-on-off-cod-activated":false,
    "is-on-off-cod-enabled":true,
    "is-on-off-cod-installed":false,
    "is-real-cbu-available":false,
    "is-secure-execution-enabled":true,
    "is-global-key-installed":true,
    "is-host-key-installed":true,
    "is-service-required":false,
    "lan-interface1-address":"00106f2371e2",
    "lan-interface1-type":"ethernet",
    "lan-interface2-address":"00106f2371e3",
    "lan-interface2-type":"ethernet",
    "last-energy-advice-time":-1,
    "last-used-activation-profile":"",
    "last-used-iocds":"A1",
    "machine-model":"T02",
    "machine-serial-number":"00000005742D",
    "machine-type":"8562",
    "msu-permanent":693,
    "msu-permanent-plus-billable":693,
    "msu-permanent-plus-temporary":693,
    "msu-purchased":693,
    "name":"SETR87",
    "network1-ipv4-alt-ipaddr":"192.0.2.0",
    "network1-ipv4-mask":"255.255.255.0",
    "network1-ipv4-pri-ipaddr":"192.0.2.0",
    "network1-ipv6-info":[
       {
          "alt-ip-address":"2001:0db8::210:6fff:fe23:749e",
          "prefix":64,
          "pri-ip-address":"2001:0db8::210:6fff:fe23:71e2",
          "type":"link-local"
       }
    ],
    "network2-ipv4-alt-ipaddr":"192.0.2.0",
    "network2-ipv4-mask":"255.255.255.0",
    "network2-ipv4-pri-ipaddr":"192.0.2.0",
    "network2-ipv6-info":[
       {
          "alt-ip-address":"2001:0db8:210:6fff:fe23:749f%eth0",
          "prefix":64,
          "pri-ip-address":"2001:0db8:210:6fff:fe23:71e3%eth0",
          "type":"link-local"
       }
    ],
```


Figure 572. Get CPC Properties: Response (Part 5)

```
    "next-activation-profile-name":"DEFAULT",
    "object-id":"d9c47445-64df-39f6-9d74-6376701508b5",
    "object-uri":"/api/cpcs/d9c47445-64df-39f6-9d74-6376701508b5",
    "on-off-cod-activation-date":0,
    "parent":null,
    "primary-host-import-key-id-pattern":
      "000102030405060708090a0b0c0e0faabbccddeeff112233445566778899ffee",
    "processor-count-aap":0,
    "processor-count-defective":0,
    "processor-count-general-purpose":0,
    "processor-count-icf":0,
    "processor-count-ifl":0,
    "processor-count-iip":0,
    "processor-count-pending":0,
    "processor-count-pending-aap":0,
    "processor-count-pending-general-purpose":0,
    "processor-count-pending-icf":0,
    "processor-count-pending-ifl":0,
    "processor-count-pending-iip":0,
    "processor-count-pending-service-assist":0,
    "processor-count-permanent-icf": 0,
    "processor-count-permanent-ifl": 0,
    "processor-count-permanent-iip": 0,
    "processor-count-permanent-service-assist": 0,
    "processor-count-service-assist":0,
    "processor-count-spare":0,
    "processor-count-unassigned-icf": 0,
    "processor-count-unassigned-ifl": 0,
    "processor-count-unassigned-iip": 0,
    "processor-count-unassigned-service-assist": 0,
    "processor-running-time":null,
    "processor-running-time-type":"system-determined",
    "se-version":"2.15.0",
    "software-model-permanent":"Z04",
    "software-model-permanent-plus-billable":"Z04",
    "software-model-permanent-plus-temporary":"Z04",
    "software-model-purchased": "Z04",
    "status":"operating",
    "storage-customer":360448,
    "storage-customer-available":355328,
    "storage-customer-central":0,
    "storage-customer-expanded":0,
    "storage-hardware-system-area":163840,
    "storage-total-installed":524288,
    "storage-vfm-increment-size":16,
    "storage-vfm-total":0,
    "zcpc-ambient-temperature":null,
    "zcpc-dew-point":null,
    "zcpc-exhaust-temperature":null,
    "zcpc-heat-load":2534,
    "zcpc-heat-load-forced-air":null,
    "zcpc-heat-load-water":null,
    "zcpc-humidity":null,
    "zcpc-maximum-potential-heat-load":47491,
    "zcpc-maximum-potential-power":13909,
    "zcpc-power-cap-allowed":null,
    "zcpc-power-cap-current":null,
    "zcpc-power-cap-maximum":null,
    "zcpc-power-cap-minimum":null,
    "zcpc-power-capping-state":null,
    "zcpc-power-consumption":742,
    "zcpc-power-rating":16628,
    "zcpc-power-save-allowed":null,
    "zcpc-power-saving":null,
    "zcpc-power-saving-state":null
}
```


Figure 573. Get CPC Properties: Response (Part 6)

Update CPC Properties

The `Update CPC Properties` operation updates one or more writable properties of the CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`group-uri`	String/ URI	Optional	The object URI for the Custom Group object to be used for updating the value of the **next-activation-profile-name** property. If omitted the system-defined Logical Partition group will be used.

Request body contents

The request body is expected to contain one or more field names representing writable CPC properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the CPC object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **System Details** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	19	The request body contains a field whose corresponding data model property is not writable on this HMC and/or SE version.
	268	The requested update requires that the **processor-running-time-type** property already contain **"user-determined"** or that the request body also requests an update of the **processor-running-time-type** property to **"user-determined"**.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Start CPC

The `Start CPC` operation starts the CPC object designated by *{cpc-id}*. The target CPC object must be enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/start
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve job status or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1051. The **job-results** field is null when this operation

is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

`Start CPC` is a process that makes a CPC operational, which means either:

- The CPC is ready to have a control program or operating system loaded, or
- The CPC has loaded and is running a control program or operating system.

`Start CPC` makes a CPC operational by:

- Using predefined information to set the operational capabilities and characteristics of the CPC
- Checking the current status of the CPC, and then performing only the operations necessary to make it operational.

So, using `Start CPC` is not limited to starting the system. Using `Start CPC` is recommended whenever you want to make the CPC or its partition objects operational.

A complete start starts the CPC and its partition objects completely in a single step. The result of a complete start is an operational CPC with partition objects loaded and running operating systems. The current status of the CPC and its partition objects determines which operations are performed during the start to make them operational. The start may include:

1. Turning CPC power on.
2. Performing a power-on reset, this includes allocating system resources to the CPC.
3. Then starting partition objects to support multiple images. Starting each partition object includes:
   a. Initializing it.
   b. Allocating system resources to it.
   c. Loading it with a control program or operating system.

Because the status of the CPC and its partition objects determines which operations must be performed during the start to make them operational, one or more operations listed above may not be performed during the start. For example:

- Starting the CPC does not perform a power-on reset if the CPC has already been power-on reset and the desired applicable settings, such as the operating mode and active input/output configuration data set (IOCDS), are already in effect.
- Starting the CPC does not perform any operations if the CPC is already operational and all desired settings are already in effect.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 1051.

This operation supports cancellation of its asynchronous processing identified by the Job URI provided in the response body. Use the `Cancel Job` operation to request cancellation. Note that it may no longer be possible to cancel the job when the cancellation request is issued. The job status and reason codes will indicate whether the job was canceled or ran to completion.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Start** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1049.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC object designated by *{cpc-id}* is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed, was canceled, or was rejected due to the current CPC status. The CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage note

This operation starts a CPC that is enabled for DPM only. If the targeted CPC object is not enabled for DPM, the `Activate CPC` operation may be used instead. Refer to "Activate CPC" on page 1054 for details.

Stop CPC

The `Stop CPC` operation stops the CPC object designated by *{cpc-id}*. The target CPC object must be enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/stop
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve stop status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1053. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

`Stop CPC` is an orderly process for shutting down and turning off the CPC.

Shutting down and turning off the CPC, referred to also as stopping the CPC, includes:

- Ending hardware and software activity
- Clearing, releasing, and de-allocating hardware resources
- Turning off power.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes seen in "Job status and reason codes" on page 1053.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Stop** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1052.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC object designated by *{cpc-id}* is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status. The CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage note

This operation stops a CPC object that is enabled for DPM only. If the targeted CPC object is not enabled for DPM, the `Deactivate CPC` operation may be used instead. Refer to "Deactivate CPC" on page 1057 for details.

Activate CPC

The `Activate CPC` operation activates the CPC object designated by *{cpc-id}*. This operation is not permitted when the CPC is enabled for DPM. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/activate
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`activation-profile-name`	String (1-16)	Optional	The name of the activation profile to be used for the request. If not provided, the request uses the profile name specified in the **next-activation-profile-name** property for the CPC object.
`force`	Boolean	Optional	Whether this operation is permitted when the CPC is in **"operating"** status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve activation status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1056. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Activation is a process that makes a CPC operational, which means either:

- The CPC is ready to have a control program or operating system loaded, or
- The CPC has loaded and is running a control program or operating system.

Activation makes a CPC operational by:

- Using predefined information, referred to as an activation profile, to set the operational capabilities and characteristics of the CPC
- Checking the current status of the CPC, and then performing only the operations necessary to make it operational as specified in the activation profile.

So, using activation is not limited to starting the system. Using activation is recommended whenever you want to make the CPC or its logical partitions operational.

A complete activation activates the CPC and its logical partitions completely in a single step. The result of a complete activation is an operational CPC with logical partitions loaded and running operating systems. The current status of the CPC and its logical partitions determines which operations are performed during activation to make them operational. Activation may include:

1. Turning CPC power on.
2. Performing a power-on reset, this includes allocating system resources to the CPC.
3. Then activating logical partitions to support multiple images. Activating each logical partition includes:
   a. Initializing it.
   b. Allocating system resources to it.
   c. Loading it with a control program or operating system.

Because the status of the CPC and its logical partitions determines which operations must be performed during activation to make them operational, one or more operations listed above may not be performed during activation. For example:

- Activating the CPC does not perform a power-on reset if the CPC has already been power-on reset and the applicable settings in its assigned activation profile, such as the operating mode and active input/output configuration data set (IOCDS), are already in effect.
- Activating the CPC does not perform any operations if the CPC is already operational and all settings in its assigned activation profile are already in effect.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 1056.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Activate** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1054.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be performed because the CPC object designated by *{cpc-id}* is currently enabled for DPM, which is not supported in this operation.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status and use of the force=false parameter. If rejected due to force=false, the CPC status is unchanged. If the operation failed, the CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage note

This operation activates a CPC object that is not enabled for DPM. If the targeted CPC object is enabled for DPM, the `Start CPC` operation may be used instead. Refer to "Start CPC" on page 1049 for details.

Deactivate CPC

The `Deactivate CPC` operation deactivates the CPC object designated by *{cpc-id}*. This operation is not permitted when the CPC is enabled for DPM. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/deactivate
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/Opt	Description
`force`	Boolean	Optional	Whether this operation is permitted when the CPC is in **"operating"** status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve activation status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1059. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Deactivation is an orderly process for shutting down and turning off the CPC.

Shutting down and turning off the CPC, referred to also as deactivating the CPC, includes:

- Ending hardware and software activity
- Clearing, releasing, and de-allocating hardware resources
- Turning off power.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151

for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes seen in "Job status and reason codes" on page 1059.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Deactivate** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1057.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be preformed because the CPC object designated by *{cpc-id}* is currently enabled for DPM, which is not supported in this operation.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current CPC status and use of the force=false parameter. If rejected due to force=false, the CPC status is unchanged. If the operation failed, the CPC status is unknown. Refer to the message parameter in the error response body for details.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Usage note

This operation deactivates a CPC object that is not enabled for DPM. If the targeted CPC object is enabled for DPM, the `Stop CPC` operation may be used instead. Refer to "Stop CPC" on page 1052 for details.

Import Profiles

The `Import Profiles` operation imports activation profiles and/or system activity profiles for the CPC from the SE hard drive into the CPC object designated by *{cpc-id}*. This operation is not permitted when the CPC is enabled for DPM. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`profile-area`	Integer (1-4)	Required	The numbered hard drive area from which the profiles are imported. Use the profile-area value specified on the prior `Export Profiles` operation.

Description

The Support Element provides four reusable areas on its hard drive from which the data save by a prior Export Profiles can be read.

Exporting and importing profiles is necessary only when you intend to have your current system and Support Element replaced with a new system and Support Element. Support Element (Version 2.12.1 and newer) information can be found on console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Export/Import Profile Data (API only)** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	This operation cannot be preformed because the CPC designated by the request URI is enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	279	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Export Profiles

The `Export Profiles` operation exports activation profiles and/or system activity profiles from the CPC object designated by *{cpc-id}* to the SE hard drive. This operation is not permitted when the CPC is enabled for DPM. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/export-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`profile-area`	Integer (1-4)	Required	The numbered hard drive area to which the profiles are exported. Any existing data is overwritten.

Description

The Support Element provides four reusable areas on its hard drive that can be used as temporary save areas. The choice of save area is up to the caller.

Exporting and importing profiles is necessary only when you intend to have your current system and Support Element replaced with a new system and Support Element. Support Element (Version 2.12.1 and newer) information can be found on the console help system. For information about earlier versions of the Support Element, see the *Support Element Operations Guide*.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Export/Import Profile Data (API only)** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	4	The operation cannot be performed because the CPC designated by the request URI is enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	279	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Set Auto-Start List

The `Set Auto-Start List` operation identifies a CPC's partitions that are automatically started when the CPC is started. It also specifies the order in which they are started. The target CPC object is designated by *{cpc-id}* and must be enabled for DPM.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-auto-start-list
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`auto-start-list`	Array of objects	Required	An array of auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. The format of that object is described in the “Class specific additional properties” on page 1013.

Description

This operation defines the CPC's partition auto-start list. The auto-start list is an ordered list of partitions and/or groups of partitions that are automatically started when the CPC is started. See the “Class specific additional properties” on page 1013 for the format of the entries in this list. Each partition or partition group is started before those that are later than it in the list. The order in which partitions within a partition group are started is not specified. An empty auto-start list indicates that no partitions are to be automatically started.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

The URI path must designate an existing CPC, and the API user must have object-access permission to it, otherwise, status code 404 (Not Found) is returned. Each Partition URI in the request body must designate an existing partition, and the API user must have object-access permission to it, otherwise, status code 404 (Not found) is returned. If the auto-start list contains duplicate partition group names or a partition is listed multiple times in the list or an empty partition group is specified, HTTP status code 400 (Bad Request) is returned.

If the CPC object designated by *{cpc-id}* is not enabled for DPM, HTTP status code 409 (Conflict) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **System Details** task.
- Object-access permission to all partitions specified in the request body.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 500. Set Auto-Start List: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	Two or more of the partition group names specified in the auto-start-list in the request body are duplicates.
	333	One or more of the partition URIs specified in the auto-start-list in the request body does not designate a partition object of the CPC object designated by *{cpc-id}*.
	334	Two or more of the partition URIs specified in the auto-start-list in the request body are duplicates.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	One or more of the partition URIs specified in the auto-start-list in the request body does not designate an existing partition object, or the API user does not have object access permission to the object.
409 (Conflict)	5	The operation cannot be performed because the CPC designated by *{cpc-id}* is currently not enabled for DPM.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/e5ae3ab6-ac8d-33bc-9739-eb142d89804d/operations/set-auto-start-
   list HTTP/1.1
x-api-session: 2kk848szmu8mo00lkj4c19254fiejeanyv316j0d5d4uppgp8t
content-type: application/json
content-length: 384
{
   "auto-start-list":[
      {
         "partition-uri":"/api/partitions/0589baec-d599-11e5-8959-42f2e9105e9b",
         "post-start-delay":15,
         "type":"partition"
      },
      {
         "description":"description for group1",
         "name":"group1",
         "partition-uris":[
            "/api/partitions/39daff36-b51e-11e5-9710-42f2e9105e9b",
            "/api/partitions/8c6e48a2-ce91-11e5-98b3-42f2e9105e9b"
         ],
         "post-start-delay":0,
         "type":"partition-group"
      }
   ]
}
```

Figure 574. Set Auto-Start List: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 17 Feb 2016 20:54:12 GMT

<No response body>
```

Figure 575. Set Auto-Start List: Response

Add Temporary Capacity

The `Add Temporary Capacity` operation adds temporary processors or increases temporary model capacity to the CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/add-temp-capacity
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`record-id`	String (1-8)	Required	Identifies the capacity record to be used for this request

Field name	Type	Rqd/Opt	Description
`software-model`	String (1-3)	Optional	The target software model. Implicit in the software model is the number of general purpose processors desired. When provided, this value must be one of the software models defined within the capacity record, and must represent a number of general purpose processors equal to or greater than the current software model. If the provided software model is equal to the current software model, or is not provided, the current software model is not changed.
`processor-info`	Array of processor-info objects	Optional	A nested object that defines the number of specialty processors to be added. If not provided, the number of specialty processors is not changed.
`force`	Boolean	Optional	Whether the operation proceeds if not enough processors are available (true) or not (false). The default is false.
`test`	Boolean	Required	Whether the request should activate real or test resources for the capacity record. Set true if test or set to false if real. This is mainly used for Capacity Backup Upgrade (CBU) activations. See the *Capacity on Demand User's Guide*. For most records, field should be set to false.

processor-info object

Field name	Type	Rqd/Opt	Description
`processor-type`	String Enum	Required	Identifies the type of specialty processors to be affected. One of: • **"aap"** - Application Assist Processor • **"ifl"** - Integrated Facility for Linux processor • **"icf"** - Internal Coupling Facility processor • **"iip"** - z Integrated Information Processors • **"sap"** - System Assist Processor
`num-processor-steps`	Integer	Optional	The delta to the current number of processors. If not provided, the number of processors is not changed.

Description

Removal of these temporary resources can be performed manually through the Remove Temporary Capacity operation or automatically upon expiration of the capacity record.

Refer to the *Capacity on Demand User's Guide* for details on temporary capacity changes.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Perform Model Conversion** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	271	A duplicate processor-type entry was found in the processor-info array, remove the duplicate entry.
	275	The test value does not match the value stored in the capacity record.
	276	Either the request specifies more resources than available or the requested software model specifies fewer resources than the current software model.
	277	A temporary capacity record is already active. It must be deactivated before a new capacity record can be activated.
	278	The software-model value was not found in the capacity record. Only software models as defined in the target capacity record can be specified.
	298	The operation parameters conflict with the capacity record type: • force=true is permitted only for CBU, CPE and loaner capacity records.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The requested capacity record does not exist.

HTTP error status code	Reason code	Description
409 (Conflict)	1	The operation is unavailable in the current CPC state: • The SE is not configured to allow temporary capacity changes through an API • The CPC **status** property is not **"operating"** or the CPC iml-mode property is not **"lpar"** • No physical processors are operating • The CPC is IMLed in a test or debug mode • An IML is required
	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	297	Some, but not all, of the requested resources were added.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Remove Temporary Capacity

The `Remove Temporary Capacity` operation removes temporary processors or decreases temporary model capacity from the CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/remove-temp-capacity
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`record-id`	String (1-8)	Required	Identifies the capacity record to be used for this request

Field name	Type	Rqd/Opt	Description
`software-model`	String (1-3)	Optional	The target software model. Implicit in the software model is the number of general purpose processors desired. When provided, this value must be one of the software models defined within the capacity record, and must represent a number of general purpose processors equal to or less than the current software model. If the provided software model is equal to the current software model, or is not provided, the current software model is not changed.
`processor-info`	Array of processor-info objects	Optional	A nested object that defines the number of specialty processors to be removed. If not provided, the number of specialty processors is not changed. Refer to "Request body contents" on page 1064 of the `Add Temporary Capacity` operation for details.

Description

When you are finished using all or part of a capacity upgrade, you can remove processors or decrease model capacity using this operation. You can only remove activated resources for the specific offering. You cannot remove dedicated processors or the last processor of a processor type.

If you remove resources back to the base configuration, the capacity record activation is completed. That is, if you remove the last temporary processor, your capacity record is deactivated. For a CBU and On/Off CoD record, to add resources again, you must use another `Add Temporary Capacity` operation. For an On/Off CoD test or CPE record, once the record is deactivated, it is no longer available for use. You can then delete the record.

After removal of the resources, the capacity record remains as an installed record. If you want a record deleted, you must manually select the record on the Installed Records page and click Delete.

Refer to the *Capacity on Demand User's Guide* for details on temporary capacity changes.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Perform Model Conversion** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	271	A duplicate processor-type entry was found in the processor-info array, remove the duplicate entry.
	276	Either the request specifies more resources than are currently active or the requested software model specifies more resources than the current software model.
	278	The software-model value was not found in the capacity record. Only software models as defined in the target capacity record can be specified.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The requested capacity record does not exist.
409 (Conflict)	1	The operation is unavailable in the current CPC state: • The SE is not configured to allow temporary capacity changes through an API • The CPC **status** property is not **"operating"** or the CPC **iml-mode** property is not **"lpar"**.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Swap Current Time Server

The `Swap Current Time Server` operation changes the role of the CPC object designated by *{cpc-id}* to the Current Time Server (CTS). This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/swap-cts
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`stp-id`	String (1-8)	Required	Identifies the STP. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).

Description

This operation changes the role of the CPC object designated by *{cpc-id}* to the Current Time Server (CTS).

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission to the `Manage System Time` and **Modify Assigned Server Roles** tasks.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	285	The CPC targeted by this operation is already the Preferred Time Server.
	286	The operation was rejected for one of the following reasons: • The **stp-id** field value does not match the current CTN identifier • The operation is not permitted for a mixed CTN.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress • no alternate is active • this CPC is not the backup time server
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Set STP Configuration

The `Set STP Configuration` operation updates the configuration for an STP-only Coordinated Timing Network. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-stp-config
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`stp-id`	String (1-8)	Required	The current STP identifier for the CTN, used to verify that the CPC is a member of the correct CTN. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).
`new-stp-id`	String (1-8)	Optional	If provided, the new STP identifier for the CTN, Can contain 0-9, a-z, A-Z, underline (_) and dash (-).
`force`	Boolean	Required	Whether a disruptive operation is allowed (true) or rejected (false)
`preferred-time-server`	stp-node object	Required	Identifies the CPC object to be the Preferred Time Server. Refer to Table 415 on page 799 for details.
`backup-time-server`	stp-node object	Optional	Identifies the CPC object to be the Backup Time Server. If not provided, the STP has no Backup Time Server. Refer to Table 415 on page 799 for details.

Field name	Type	Rqd/Opt	Description
`arbiter`	stp-node object	Optional	Identifies the CPC object to be the Arbiter for the CTN. If not provided, the STP has no Arbiter. Refer to Table 415 on page 799 for details.
`current-time-server`	String Enum	Required	Identifies the role of the Current Time Server (CTS). One of: • **"preferred"** - the Preferred Time Server is the CTS • **"backup"** - the Backup Time Server is the CTS.

Description

The CPC object designated by *{cpc-id}* must be the system that becomes the Current Time Server (CTS).

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission to the `Manage System Time` and **Modify Assigned Server Roles** tasks.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	282	The operation was rejected, due to an incomplete preferred, backup or arbiter nested object specification. Refer to Table 415 on page 799 for details.
	284	This operation does not target the Current Time Server CPC.
	285	The operation was rejected, due to one of the following configuration errors: • A **backup-time-server** object is required when providing an **arbiter** object • A **backup-time-server** object is required when **current-time-server** is backup • The **preferred-time-server**, **backup-time-server** and **arbiter** objects do not reference different CPCs
	286	The operation was rejected for one of the following reasons: • The **stp-id** field value does not match the current CTN identifier • The operation is not permitted for a mixed CTN
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	An object URI in one of the **stp-node** objects in the request body does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress • no alternate is active • the operation is not permitted for a mixed-CTN.
	287	The provided configuration can only be set by specifying force=true.
	288	No communication path between preferred-time-server and backup-time-server.
	289	No communication path to the arbiter.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during the operation.

HTTP error status code	Reason code	Description
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Change STP-only Coordinated Timing Network

The `Change STP-only Coordinated Timing Network` operation, sent to the CPC object designated by *{cpc-id}* with the role of Current Time Server (CTS) in an STP-only Coordinated Timing Network (CTN), changes the STP ID portion of the CTN ID for the entire STP-only CTN. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/change-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`stp-id`	String (1-8)	Required	The new STP identifier. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).

Description

This operation, sent to the CPC object designated by *{cpc-id}* with the role of Current Time Server (CTS) in an STP-only Coordinated Timing Network (CTN), changes the STP ID portion of the CTN ID for the entire STP-only CTN.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission to the `Manage System Time` and **Rename CTN** tasks.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	284	The CPC targeted by the operation is not the Current Time Server. Retry the operation using the object-uri for the Current Time Server CPC.
	286	The operation is not permitted for a mixed CTN.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • an ETR reverse migration is in progress
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Join STP-only Coordinated Timing Network

The `Join STP-only Coordinated Timing Network` operation allows a CPC object designated by *{cpc-id}* to join an STP-only Coordinated Timing Network (CTN). This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/join-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`stp-id`	String (1-8)	Required	Identifies the STP to be joined. Can contain 0-9, a-z, A-Z, underline (_) and dash (-).

Description

If the CPC object is already participating in a different STP-only CTN and is the Current Time Server (CTS), the operation is rejected. Otherwise, the CPC object is removed from its current CTN and joins the specified CTN.

If the CPC object has an ETR ID, the ETR ID is removed.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission to the `Manage System Time` and **Add Systems to CTN** tasks.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Leave STP-only Coordinated Timing Network

The `Leave STP-only Coordinated Timing Network` operation allows a CPC object designated by *{cpc-id}* to leave the STP-only Coordinated Timing Network (CTN) in which it currently participates. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/leave-stponly-ctn
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Description

The CPC object cannot be the Current Time Server (CTS) in the CTN in which it is currently participating.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission to the `Manage System Time` and **Remove Systems from CTN** tasks.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	286	The operation is not permitted for a mixed CTN.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
409 (Conflict)	1	The requested operation cannot be performed, due to the state of the object: • Server Time Protocol is not enabled on this CPC • this CPC is not a member of a CTN • this CPC is the Current Time Server.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	272	An unexpected error occurred during processing of the Server Time Protocol operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Get CPC Audit Log

The `Get CPC Audit Log` operation returns the CPC's audit log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-audit-log
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

Description

This operation returns the CPC's audit log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the CPC itself. If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and that CPC must be at a release level that supports this operation. If any of these conditions is not met, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have action/task permission to the **Audit and Log Management** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the audit log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented on console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object specified in the request URI
- For the web services interface, action/task permission to the **Audit and Log Management** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 1079.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 501. Get CPC Audit Log: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	307	The request timed out while attempting to communicate with the SE or while attempting to get the log data.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/operations/get-audit-log
   HTTP/1.1
x-api-session: 1ui4gmb59aunfkk8of69nrpks5mtnq5xjc613rdxjq53iv0e1j
```

Figure 576. Get CPC Audit Log: Request

```
200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Fri, 03 Oct 2014 00:10:35 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items":[],
      "event-details":[
         {
            "event-details-data-items":[],
            "event-details-message":"S32 - Disabled"
         }
      ],
      "event-id":"1809",
      "event-message":"Power Cap settings have changed.",
      "event-name":"POWERCAP",
      "event-time":1412202863030,
      "user-uri":null,
      "userid":null
   },
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"HMCCHGM(1.2.3.5)"
         },
         {
            "data-item-number":1,
            "data-item-type":"string",
            "data-item-value":"S32"
         }
      ],
      "event-details":[],
      "event-id":"734",
      "event-message":"Remote support call generated on S32 is being handled by
         call-home server HMCCHGM(1.2.3.5).",
      "event-name":"TRSF_OFFER",
      "event-time":1412262459470,
      "user-uri":null,
      "userid":null
   }
]
```

Figure 577. Get CPC Audit Log: Response

Get CPC Security Log

The `Get CPC Security Log` operation returns the CPC's security log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-security-log
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

Description

This operation returns the CPC's security log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the CPC itself. If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and that CPC must be at a release level that supports this operation. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have action/task permission to the **View Security Logs** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented on console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC object specified in the request URI
- For the web services interface, action/task permission to the **View Security Logs** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1082.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 502. Get CPC Security Log: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	307	The request timed out while attempting to communicate with the SE or while attempting to get the log data.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/operations/get-security-log
   HTTP/1.1
x-api-session: 1ui4gmb59aunfkk8of69nrpks5mtnq5xjc613rdxjq53iv0e1j
```

Figure 578. Get CPC Security Log: Request

```
200 OK
server: zSeries management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Fri, 03 Oct 2014 00:10:35 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items":[],
      "event-details":[],
      "event-id":"778",
      "event-message":"Mirroring data from the primary Support Element to the
         alternate Support Element started.",
      "event-name":"ASEMIRRST",
      "event-time":1412258403480,
      "user-uri":null,
      "userid":null
   },
   {
      "event-data-items":[
         {
            "data-item-number":0,
            "data-item-type":"string",
            "data-item-value":"Communications to the Alternate SE was not
               active."
         }
      ],
      "event-details":[],
      "event-id":"779",
      "event-message":"Mirroring data from the primary Support Element to the
         alternate Support Element failed. Communications to the Alternate SE was not
         active.",
      "event-name":"ASEMIRRNO",
      "event-time":1412258414110,
      "user-uri":null,
      "userid":null
   }
]
```

Figure 579. Get CPC Security Log: Response

Get CPC Events Log

The `Get CPC Events Log` operation returns the console events log, filtered according to the query parameters, if specified. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-events-log
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created earlier than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter log entries. Entries created later than this time are omitted from the results. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Name	Type	Rqd/Opt	Description
`event-id`	String	Optional	A regular expression used to limit returned entries to those that have a matching event ID. This query parameter can be used to limit the data returned to event IDs that are desired. If not specified, then no such filtering is performed.
`max-entries`	Integer	Optional	An integer value greater than zero that indicates the maximum number of entries to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the timestamp of the last entry returned as the begin-time on a subsequent invocation of this operation can get the next set of entries. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON array of JSON objects. For the web services interface the response is returned using HTTP chunked transfer encoding, while for the BCPii interface it is not. Each array element is a log-entry-info object containing information about a single log entry. The array elements are in order of increasing timestamp. See Table 443 on page 825 for more information.

Description

This operation returns the Console's events log in increasing timestamp order, filtered according to the query parameters, if specified. Each log entry pertains to a specific event that occurred on or to a managed object or the console itself. The log entries can be limited by specifying explicit filtering criteria on the request. If the begin-time query parameter is specified, then any entries earlier than that time are omitted. If the end-time query parameter is specified, then any entries later than that time are omitted. If the **event-id** query parameter is specified, then any entries with an event ID that does not match are omitted. If the **max-entries** query parameter is specified, then the number of returned entries will not exceed this value.

For the web services interface the API user must have action/task permission to the **View Console Events** task; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned.

On successful execution, the response body contains an array of filtered log entries. If the security log is empty or there are no entries to be returned after filtering, then an empty array is provided. Each log entry contains the event ID, event name and event message. If there are data items included in the event message, they are available separately. The order and meaning of the substitution items for each event ID are documented in the console help system in the HMC Introduction topic **Audit, Event, and Security Log Messages**.

Authorization requirement

This operation has the following authorization requirements:

- For the web services interface, action/task permission to the **View Console Events** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described "Response body contents" on page 1085.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/f7f2ded1-75dc-3826-9927-0ffa94e22806/operations/get-events-log
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
```

Figure 580. Get CPC Events Log: Request

```
200 OK
server: Hardware management console API web server / 2.0
transfer-encoding: chunked
cache-control: no-cache
date: Sat, 06 Mar 2021 19:19:44 GMT
content-type: application/json;charset=ISO-8859-1
[
   {
      "event-data-items": [
         {
            "data-item-number": 0,
            "data-item-type": "string",
            "data-item-value": "sysprog"
         },
         {
            "data-item-number": 1,
            "data-item-type": "long",
            "data-item-value": 0
         },
         {
            "data-item-number": 2,
            "data-item-type": "long",
            "data-item-value": 0
         },
         {
            "data-item-number": 3,
            "data-item-type": "string",
            "data-item-value": ""
         },
         {
            "data-item-number": 4,
            "data-item-type": "string",
            "data-item-value": "3"
         },
         {
            "data-item-number": 5,
            "data-item-type": "string",
            "data-item-value": "Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/
60.0"
         }
      ],
      "event-details": [],
      "event-id": "1408",
      "event-message": "User sysprog has logged on from the console to session id 3.",
      "event-name": "Logon",
      "event-time": 1615060019180,
      "user-uri": "/api/users/8efd81be-7c6a-11eb-ae1e-fa163e1cb540",
      "userid": "SYSPROG"
   },
   {
      "event-data-items": [
         {
            "data-item-number": 0,
            "data-item-type": "string",
            "data-item-value": "\"View Console Tasks Performed\""
         },
         {
            "data-item-number": 1,
            "data-item-type": "string",
            "data-item-value": "5"
         },
         {
            "data-item-number": 2,
            "data-item-type": "string",
            "data-item-value": "sysprog"
        },
```

Figure 581. Get CPC Events Log: Response (Part 1)

```
         {
            "data-item-number": 3,
            "data-item-type": "string",
            "data-item-value": "3"
         },
         {
            "data-item-number": 4,
            "data-item-type": "string",
            "data-item-value": ""
         }
      ],
      "event-details": [],
      "event-id": "1989",
      "event-message": "Task \"View Console Tasks Performed\" with identifier 5 started by user
sysprog in session 3.",
      "event-name": "TaskStart",
      "event-time": 1615060035000,
      "user-uri": "/api/users/8efd81be-7c6a-11eb-ae1e-fa163e1cb540",
      "userid": "SYSPROG"
   },
   {
      "event-data-items": [
         {
            "data-item-number": 0,
            "data-item-type": "string",
            "data-item-value": "\"View Console Tasks Performed\""
         },
         {
            "data-item-number": 1,
            "data-item-type": "string",
            "data-item-value": "5"
         },
         {
            "data-item-number": 2,
            "data-item-type": "string",
            "data-item-value": "sysprog"
         }
      ],
      "event-details": [],
      "event-id": "1991",
      "event-message": "Task \"View Console Tasks Performed\" with identifier 5 for user
sysprog has ended.",
      "event-name": "TaskEnd",
      "event-time": 1615060038260,
      "user-uri": "/api/users/8efd81be-7c6a-11eb-ae1e-fa163e1cb540",
      "userid": "SYSPROG"
   }
]
```


Figure 582. Get CPC Events Log: Response (Part 2)

List CPC Hardware Messages

The `List CPC Hardware Messages` operation lists the current set of hardware messages associated with the CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages
```

In this request, the URI variable *{cpc-id}* is the object ID of a CPC object for which hardware messages are to be listed.

Query Parameters

Name	Type	Rqd/Opt	Description
`begin-time`	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created earlier than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.
`end-time`	Timestamp	Optional	A timestamp used to filter hardware messages. Messages created later than this time are omitted from the results. The value is specified as the number of milliseconds since the epoch and must be greater than or equal to 0. If not specified, then no such filtering is performed.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`hardware-messages`	Array of hardware-message-info objects	Array of nested hardware-message-info objects as defined in the next table.

Each nested hardware-message-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The canonical URI path of the hardware message. The URI is in the following form: `/api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}`
`timestamp`	Timestamp	The date and time the hardware message was created
text	String	The text of the hardware message.

Description

This operation returns a set of CPC hardware messages in increasing timestamp order, filtered according to the query parameters, if specified. Each hardware message describes an event or notification that may require the operator's attention. The list of hardware messages can be limited by specifying explicit filtering criteria on he request.

If the **begin-time** query parameter is specified, then any entries earlier than that time are omitted. If the **end-time** query parameter is specified, then any entries later than that time are omitted.

If there are no hardware messages associated with the CPC, or if no hardware messages are to be included in the results due to filtering, an empty array is returned and the operation completes successfully.

The URI path must designate an existing CPC and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the `Hardware Messages` task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1089.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 503. List CPC Hardware Messages: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The **begin-time** value is greater than the **end-time** value.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the `Hardware Messages` task or the **Hardware Messages** task in view-only mode.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/hardware-messages HTTP/1.1
x-api-session: 35rr3x40qbnou8zwmx8ad80ldp8koes4f2abcl6m1fg5jm4tug
```

Figure 583. List CPC Hardware Messages: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 06 Oct 2014 17:08:58 GMT
content-type: application/json;charset=UTF-8
content-length: 242
{
   "hardware-messages":[
      {
         "element-uri":"/api/cpcs/1946e00f-401b-3aa6-84a3-5e49614743ec/hardware-messages/
            43a2922a-4d6b-11e4-972f-42f2e9ccd169",
         "text":"Licensed internal code has detected a problem.   [Problem # 49]",
         "timestamp":1412608381950
      }
   ]
}
```

Figure 584. List CPC Hardware Messages: Response

Get CPC Hardware Message Properties

The `Get CPC Hardware Message Properties` operation retrieves the properties of a single CPC hardware message. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message to retrieve.

Response body contents

On successful completion, the response body contains a JSON object that provides the current values of the properties for the CPC hardware message object as defined in "Data model" on page 1010. Field names and data types in the JSON object are the same as the property names and data types defined in the data model.

Description

This operation retrieves the properties of a single CPC hardware message specified by *{hardware-message-id}*.

The URI path must designate an existing CPC, and the API user must have object access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the `Hardware Messages` task or the **Hardware Messages** task in view-only mode; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*

- Action/Task permission to the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1091.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 504. Get CPC Hardware Message Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the **Hardware Messages** task or the **Hardware Messages** task in view-only mode.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI *{hardware-message-id}* does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
   9faa758e-58b8-11ea-9fb3-00106f23f56e HTTP/1.1
x-api-session: 4o94wl43xtdv4bi0c0hu1aict0xakizcvnd5umlykpn8cmkrcl
```

Figure 585. Get CPC Hardware Message Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 18:58:32 GMT
content-type: application/json;charset=UTF-8
content-length: 849
{
   "class":"hardware-message",
   "details":{
      "corrective-action":[
         "Problem Analysis is now complete.",
         "Service is required."
      ],
      "created":1582735965261,
      "description":[
         "A power failure has been detected.",
         "The system is still operating."
      ],
      "lir-node-data":null,
      "problem-data":[
         {
            "caption":"System name",
            "value":"T204"
         },
         {
            "caption":"Date",
            "value":"Feb 26, 2020"
         },
         {
            "caption":"Time",
            "value":"11:52:45 AM"
         }
      ],
      "repair-impact":[
         "The repair of this problem can most likely be performed concurrent with
            system operations."
      ],
      "type":"common-problem"
   },
   "element-id":"9faa758e-58b8-11ea-9fb3-00106f23f56e",
   "element-uri":"/api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
      9faa758e-58b8-11ea-9fb3-00106f23f56e",
   "parent":"/api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a",
   "service-supported":true,
   "text":"Power problem.   [Problem # 165]",
   "timestamp":1582736056410
}
```

Figure 586. Get CPC Hardware Message Properties: Response

Delete CPC Hardware Message

The `Delete CPC Hardware Message` operation deletes a single CPC hardware message. This operation is supported using the BCPii interface.

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message to delete.

Description

This operation deletes a specific CPC hardware message. The hardware message to be deleted is identified by the *{hardware-message-id}* variable in the URI.

The URI path must designate an existing CPC and the API user must have object-access permission to it; otherwise status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 505. Delete CPC Hardware Message: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have Object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI *{hardware-message-id}* does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/0fdde999-5957-3129-99aa-b6f4bfbbc071/hardware-messages/
   d5591a80-43f8-11e4-ac52-42f2e910664b HTTP/1.1
x-api-session: c8un3odpy8yyp150o3poz1ud4gwyfodlwyq495327bpyn2p0z
```

Figure 587. Delete CPC Hardware Message: Request

```
204 No Content
date: Mon, 09 Feb 2015 20:07:31 GMT
server: zSeries management console API web server / 2.0

<No response body>
```

Figure 588. Delete CPC Hardware Message: Response

Request CPC Service

The `Request CPC Service` operation electronically transmits problem information to request service for the error and deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/request-service
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message.

Request body contents

An optional request body can be specified as a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`customer-name`	String	Optional	The name of the person that can be contacted about the problem.
`customer-phone`	String	Optional	The telephone number of the person that can be contacted about the problem.

Description

This operation electronically requests service for the problem reported by this hardware message. Customer contact information may optionally be provided in the request body. Upon successful completion, the service request is prepared and queued for transmission, and this hardware message is deleted.

The URI path must designate an existing CPC, and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. Remotely requesting service must also be enabled and configured on the

HMC; otherwise, status code 409 (Conflict) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a CPC hardware message that does not support service.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI *{hardware-message-id}* does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	341	The HMC is not enabled and configured for remotely requesting service.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
   9faa758e-58b8-11ea-9fb3-00106f23f56e/operations/request-service HTTP/1.1
x-api-session: 440mepm56w27o7965y61kbkm4g422hja984lrm74mc2wdpf3vw
content-type: application/json
content-length: 56
{
   "customer-name":"Jenny",
   "customer-phone":"867-5309"
}
```

Figure 589. Request CPC Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:00:06 GMT

<No response body>
```

Figure 590. Request CPC Service: Response

Get CPC Service Request Information

The `Get CPC Service Request Information` operation returns problem information and a telephone number to be used for requesting service for the error and optionally deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/get-service-
information
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message.

Query parameters

Name	Type	Rqd/Opt	Description
`delete`	Boolean	Optional	A value of **true** will delete the hardware message upon successful completion, **false** will not delete the hardware message. Default: **true**

Response body contents

On successful completion, the response body contains a JSON object with the following fields.

Name	Type	Description
service-phone	String	The telephone number to call for service.
machine-model	String (1-3)	The model of the machine where the problem occurred.

Name	Type	Description
machine-type	String (1-4)	The type of the machine where the problem occurred.
machine-serial-number	String (1-12)	The serial number of the machine where the problem occurred.
problem-type	String	The type of the problem to be reported to service.
problem-number	Integer	The identifying number of the problem to be reported to service.
problem-data	String	Additional problem data to be reported to service.
reference-code	String	The problem reference code to be reported to service.
customer-name	String	The name of the administrator of the CPC. This field will be omitted if the API user does not have authority to the **Customer Information** task.
customer-phone	String	The telephone number of the administrator of the CPC. This field will be omitted if the API user does not have authority to the **Customer Information** task.

Description

This operation is used to manually request service for the problem reported by this hardware message. This may be used if remote service is not configured or not functioning to call a service representative directly and provide the problem details. Upon successful completion, problem details to be reported to service are returned, and if the **delete** query parameter is **true**, this hardware message is deleted.

The URI path must designate an existing CPC, and the APIU user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 1097.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a CPC hardware message that does not support service.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI *{hardware-message-id}* does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
   ea2ec24e-5963-11ea-9fb3-00106f23f56e/operations/get-service-information HTTP/1.1
x-api-session: 34w3jublxvx3d51s5x0xc69kyuwshfvxy6r94u8chwkzbu06qd
```

Figure 591. Get CPC Service Request Information: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:01:45 GMT
content-type: application/json;charset=UTF-8
content-length: 231
{
   "machine-model":"T02",
   "machine-serial-number":"0000200273D8",
   "machine-type":"8562",
   "problem-data":"00L5201,1,01KU212,1",
   "problem-number":167,
   "problem-type":"1",
   "reference-code":"30C02011-6863670B",
   "service-phone":"1-800-IBM-SERV"
}
```

Figure 592. Get CPC Service Request Information: Response

Decline CPC Service

The `Decline CPC Service` operation declines service for the error and deletes the hardware message designated by the URI path. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/hardware-messages/{hardware-message-id}/operations/decline-service
```

URI Variables:

Variable	Description
{cpc-id}	Object ID of the CPC object.
{hardware-message-id}	Element ID of the hardware message.

Description

This operation is used to decline service for the problem reported by this hardware message. Upon successful completion, the hardware message is deleted.

The URI path must designate an existing CPC, and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

The URI path must designate an existing hardware message; otherwise, status code 404 (Not Found) is returned. The hardware message's **service-supported** property must also be **true**; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have Action/Task permission to the **Hardware Messages** task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **Hardware Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
	340	The element ID in the URI *{hardware-message-id}* designates a CPC hardware message that does not support service.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have Action/Task permission for the **Hardware Messages** task.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The CPC designated by the request URI does not support this operation.
	322	The element ID in the URI *{hardware-message-id}* does not designate an existing CPC hardware message.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/a0f43b73-d5aa-37dc-bf93-c8ab35ce607a/hardware-messages/
   ed6a4052-58ad-11ea-9fb3-00106f23f56e/operations/decline-service HTTP/1.1
x-api-session: 69u9tm62kmcim81nxvi556utt8bxq7lkh80jw3ayhql2p6i3b6
content-type: application/json
```

Figure 593. Decline CPC Service: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 03 Mar 2020 19:02:52 GMT

<No response body>
```

Figure 594. Decline CPC Service: Response

Export WWPN List

The `Export WWPN List` operation exports the worldwide port names (WWPNs) of the HBAs of the specified partitions.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/export-port-names-list
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC that contains all the specified partitions.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`partitions`	Array of String/ URI	Required	Array of canonical URI paths, one for each partition whose list of WWPNs is to be exported.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`wwpn-list`	String	The comma separated list contains the WWPNs along with each associated partition **name**, **adapter-id**, and **device-number**.

Description

The `Export WWPN List` operation returns the list of host port WWPNs of the partitions specified by the **partitions** field of the request body. These partitions must be part of the CPC specified by *{cpc-id}*. The list is provided in a JSON object as a single string in Comma-Separated Values (CSV) format. Each line in the string will be represented as: Partition name,Adapter ID,device-number,WWPN.

On successful execution, the WWPN list for all the HBAs defined in the specified partitions are provided in the response body, and HTTP status code 200 (OK) is returned.

The request URI path must designate an existing CPC object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. If the array of partition URIs is empty, status code 400 (Bad Request) is returned. The URIs in the request body must designate existing Partition objects and API user must have object-access permission to them; otherwise, status code 404 (Not Found) is returned. The partitions must be part of the specified CPC; otherwise, status code 400 (Bad Request) is returned. In addition, the API user must have action access permission to the **Export WWPNs** task; otherwise, status code 403 (Forbidden) is returned.

The request body is validated against the schema described in "Request body contents" on page 1102. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

If the CPC has the **"dpm-storage-management"** feature enabled, 409 (Conflict) status code is returned.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object specified in the request URI.
- Object-access permission to the partition objects specified in the request body.
- Action/task permission to the **Export WWPNs** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1102

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 506. Export WWPN List: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation.
	149	The partitions array is empty.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	The object ID *{partition-id}* does not designate an existing Partition object, or the API user does not have object-access permission to it.
409 (Conflict)	12	The operation is not supported when the **"dpm-storage-management"** feature is enabled on the CPC.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/51342eec-1ba0-3866-b639-a99860807b89/operations/export-port-names-
  list HTTP/1.1
x-api-session: 3ud8oxm0p8d8um2r3y2pjhbws8ishkf4r4vqqocrsrb1e0h0y2
content-type: application/json
content-length: 72
{
   "partitions":[
      "/api/partitions/4e12c87e-c8b2-11e5-97e4-020000000192"
   ]
}
```

Figure 595. Export WWPN List: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:59:04 GMT
content-type: application/json;charset=UTF-8
content-length: 312
{
   "wwpn-list":"#Version: 1\r\n#Partition Name,AdapterPortId,DeviceNumber,WWPN\r
      \nMyPartition_1,159,1007,0000000000000009\r\nMyPartition_1,158,1002,
      0000000000000006\r\nMyPartition_1,159,1001,0000000000000005\r\nMyPartition_
      1,158,1003,0000000000000007\r\nMyPartition_1,159,1005,000000000000000A\r\n"
}
```

Figure 596. Export WWPN List: Response

Import DPM Configuration

The `Import DPM Configuration` operation imports DPM objects such as partitions, NICs, HBAs, virtual functions, and their properties. It also restores DPM-specific CPC properties, like the description, auto-start partition list and capacity groups.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-dpm-config
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`cpc-properties`	cpc-info object	Optional	An object containing the names of CPC properties and the values to which each of those properties is to be set. This is a very limited set of CPC properties relevant to CPCs enabled for DPM mode.
`se-version`	String (1-8)	Required	The internal code release level of the primary SE on the machine from which the configuration was exported.
`available-features-list`	Array of cpc-feature-info objects	Required	The list of optional features or behaviors supported by the CPC on the machine from which the configuration was exported. If the CPC has no optional features, then the array must be empty.
`available-api-features-list`	Array of String	Optional	The API features available on the source system, as returned by the List CPC API Features operation, issued when the configuration information was exported.
`preserve-uris`	Boolean	Optional	Controls whether object IDs / element IDs are preserved or if new object IDs / element IDs will be generated. Because those IDs are components of URIs, this field controls whether the URIs of imported objects are preserved. This applies to all objects that will be recreated, for example Partitions, HBAs, NICs, Virtual Functions, Capacity Groups, etc. **Note:** This does not apply to existing objects, such as Adapters. Default: false
`preserve-wwpns`	Boolean	Optional	Controls whether HBA WWPNs are preserved or if new WWPNs will be generated. Default: false
`adapter-mapping`	Array of adapter-mapping-info objects	Optional	Array of adapter mapping information objects. Required when the I/O adapter configuration on this machine differs from that of the machine from which the configuration was exported.

Field name	Type	Rqd/Opt	Description
`partitions`	Array of objects	Required	Array of objects containing the properties of the Partitions to be imported. Each element of this array is expected to be the equivalent of a response body from a Get Partition Properties operation issued when the configuration information was exported.
`nics`	Array of objects	Optional	Array of objects containing the properties of the NICs to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get NIC Properties` operation issued when the configuration information was exported.
`hbas`	Array of objects	Optional	Array of objects containing the properties of the HBAs to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get HBA Properties` operation issued when the configuration information was exported.
`virtual-functions`	Array of objects	Optional	Array of objects containing the properties of the Virtual Functions to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Virtual Function Properties` operation issued when the configuration information was exported.
`adapters`	Array of objects	Required	Array of objects containing the properties of the Adapters in the exported configuration. Each element of this array is expected to be the equivalent of a response body from a `Get Adapter Properties` operation issued when the configuration information was exported. This will not recreate any Adapters; it is used for internal purposes only.
`virtual-switches`	Array of objects	Optional	Array of objects containing the properties of the Virtual Switches in the exported configuration. Each element of this array is expected to be the equivalent of a response body from a `Get Virtual Switch Properties` operation issued when the configuration information was exported. This will not recreate any Virtual Switches; it is used for internal purposes only.
`capacity-groups`	Array of objects	Optional	Array of objects containing the properties of the Capacity Groups to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Capacity Group Properties` operation issued when the configuration information was exported.
`storage-sites`	Array of objects	Optional	Array of objects containing the properties of the Storage Sites to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Site Properties` operation issued when the configuration information was exported.
`storage-subsystems`	Array of objects	Optional	Array of objects containing the properties of the Storage Subsystems to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Subsystem Properties` operation issued when the configuration information was exported.

Field name	Type	Rqd/Opt	Description
`storage-fabrics`	Array of objects	Optional	Array of objects containing the properties of the Storage Fabrics to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Fabric Properties` operation issued when the configuration information was exported.
`storage-switches`	Array of objects	Optional	Array of objects containing the properties of the Storage Switches to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Switch Properties` operation issued when the configuration information was exported.
`storage-control-units`	Array of objects	Optional	Array of objects containing the properties of the Storage Control Units to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Control Unit Properties` operation issued when the configuration information was exported.
`storage-paths`	Array of objects	Optional	Array of objects containing the properties of the Storage Paths to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Path Properties` operation issued when the configuration information was exported.
`storage-groups`	Array of objects	Optional	Array of objects containing the properties of the Storage Groups to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Group Properties` operation issued when the configuration information was exported.
`storage-volumes`	Array of objects	Optional	Array of objects containing the properties of the Storage Volumes to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Volume Properties` operation issued when the configuration information was exported.
`storage-templates`	Array of objects	Optional	Array of objects containing the properties of the Storage Templates to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Template Properties` operation issued when the configuration information was exported.
`storage-template-volumes`	Array of objects	Optional	Array of objects containing the properties of the Storage Template Volumes to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Template Volume Properties` operation issued when the configuration information was exported.
`virtual-storage-resources`	Array of objects	Optional	Array of objects containing the properties of the Virtual Storage Resources to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Virtual Storage Resources Properties` operation issued when the configuration information was exported.

Field name	Type	Rqd/Opt	Description
`storage-ports`	Array of objects	Optional	Array of objects containing the properties of the Storage Ports to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Storage Port Properties` operation issued when the configuration information was exported.
`network-ports`	Array of objects	Optional	Array of objects containing the properties of the Network Ports to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Network Port Properties` operation issued when the configuration information was exported.
`partition-links`	Array of objects	Optional	Array of objects containing the properties of the Partition Links to be imported. Each element of this array is expected to be the equivalent of a response body from a `Get Partition Link Properties` operation issued when the configuration information was exported.
`certificates`	Array of objects	Optional	Array of objects containing the properties of the Certificate objects to be imported and possibly assigned to one or more Partitions. Each element of this array is expected to be the equivalent of a response body from a `Get Certificate Properties` operation enriched with the fields of the corresponding response body from a `Get Encoded Certificate` operation, both issued when the configuration information was exported. [Added by feature **secure-boot-with-certificates**]

The cpc-info nested object contains the following fields:

Table 507. cpc-info nested object

Name	Type	Description
description	String (1-1024)	The descriptive text associated with this CPC object
auto-start-list	Array of auto-start-entry objects	An array of auto-start-entry objects in sequence, each representing a single partition or a group of partitions that are automatically started when this CPC is started. The format of that object is described in the "Class specific additional properties" on page 1013.

The adapter-mapping-info nested object contains the following fields:

Table 508. adapter-mapping-info nested object

Name	Type	Description
new-adapter-id	String (3)	The hexadecimal value of the adapter ID (PCHID) on this machine
old-adapter-id	String (3)	The hexadecimal value of the adapter ID (PCHID) on the old machine

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Name	Type	Description
`output`	Array of output-details objects	An array of output-details objects containing detailed information about the parts of the configuration that were not restored.

Each nested output-details info object contains the following fields:

Name	Type	Description
`category`	String/ Enum	One of the following, describing the category of the message: • **"system-to-switches-not-restored"** - Certain storage switches connections could not be restored. The **items** field contains information describing those connections. • **"acc-vf-not-restored"** - Accelerator virtual functions for certain partitions were not restored. The **items** field contains the names of those partitions. • **"partition-id-reset"** - The Partition ID (**partition-id** property) for certain partitions was changed. The **items** field contains the names of those partitions. • **"boot-type-reset"** - The boot type (**boot-device** property) for certain partitions was reset from "**iso-image**" to **"none"**. The **items** field contains the names of those partitions. • **"ssc-password-reset"** - The password (**ssc-master-pw** property) for certain SSC partitions was reset. The **items** field contains the names of those partitions. • **"boot-password-reset"** - The FTP password (**boot-ftp-password** property) for certain partitions was reset. The **items** field contains the names of those partitions. • **"processor-type-not-available"** - Certain partitions were not restored because the required processor type is not available. The **items** field contains the names of those partitions. • **"virtual-switch-port-changed"** - Certain virtual switch ports have been changed to the default port. The **items** field contains information identifying those ports.
`items`	Array of String	Additional details about which resources were modified or not restored. The value of the **category** field defines the contents of this field.

Description

This operation restores a full DPM configuration with all its artifacts like partitions, HBAs, NICs, Accelerators, and Crypto devices. Unique identifiers like Object IDs and WWPNs are preserved. This task is mainly intended for migrating a DPM configuration from an older machine to a new machine. When migrating configurations between machines of the same generation, you have to ensure consistent feature enablement settings. Migrating from a newer machine generation to an older one is not supported. The operation stops on the first fatal error. Non-fatal errors do not stop the import and provide information about unrestored objects within the response body. The request body may contain the same set of properties as generated by the `Get Inventory` operation.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission to the **Import Dynamic Partition Manager Configuration** task.
- Object-access permission to Secure Boot Certificate objects (only applies when the request body contains one or more secure boot Certificate objects to be assigned to Partitions). [Added by feature **secure-boot-with-certificates**]
- Action/task permission for the **Import Secure Boot Certificates** task (only applies when the request body contains one or more Certificate objects). [Added by feature **secure-boot-with-certificates**]
- Action/task permission for the **Assign Secure Boot Certificates** task (only applies when the request body contains one or more secure boot Certificate objects to be assigned to Partitions). [Added by feature **secure-boot-with-certificates**]

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided. If not all parts of the configuration could be restored, a 200 (OK) is returned and the response body provides additional details.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

In contrast to other operations, the following table for HTTP status and reason codes only covers key basic error conditions, as it is not feasible to track all potential error cases when importing a potentially large DPM configuration into a new system (that might already contain DPM configuration objects). In general, any error response that corresponding operations to create/attach configuration elements might emit are possible here.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The Firmware Feature enablement of the designated CPC does not support the provided configuration objects. The API features availability on the designated CPC or Console does not support the provided configuration objects.
	100	The provided **se-version** is not supported for the import operation of this CPC
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI *{cpc-id}* does not designate an existing CPC object, or the API user does not have object-access permission to it.
	4	The SE associated with the CPC designated by the request URI is not on the required code level to support this operation.
409 (Conflict)	1	The state of the CPC is not valid to perform the operation (must be in one of the following states: **"active"**, **"service-required"**, **"degraded"**, or **"exceptions"**
	5	The operation cannot be performed because the CPC designated by the URI is not in DPM mode.
	8	An object with the same URI as one in the request body already exists.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/46a42974-59b1-3574-bade-0d2cc3e2f12c/operations/import-dpm-config HTTP/1.1
x-api-session: 1bj062lh2pj8kjinq9sntoytdimuli2edkwwom7vpw0cntdh8j
content-type: application/json
content-length: 5791
{
   "adapter-mapping":[
      {
         "new-adapter-id":"1dd",
         "old-adapter-id":"100"
      }
   ],
   "adapters":[
      {
         "adapter-family":"hipersockets",
         "adapter-id":"7c0",
         "allowed-capacity":12288,
         "channel-path-id":"09",
         "class":"adapter",
         "configured-capacity":81,
         "description":"",
         "detected-card-type":"hipersockets",
         "maximum-total-capacity":12288,
         "maximum-transmission-unit-size":8,
         "name":"hipersocket",
         "network-port-uris":[
            "/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb/network-ports/0"
         ],
         "object-id":"bc5c79e6-354f-11e7-911e-00106f0d81cb",
         "object-uri":"/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb",
         "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
         "physical-channel-status":"operating",
         "port-count":1,
         "state":"online",
         "status":"active",
         "type":"hipersockets",
         "used-capacity":27
      },
```

Figure 597. Import DPM Configuration: Request (Part 1)

```
      {
         "adapter-family":"ficon",
         "adapter-id":"100",
         "allowed-capacity":64,
         "card-location":"Z22B-D101-J.01",
         "channel-path-id":"0d",
         "class":"adapter",
         "configured-capacity":10,
         "description":"",
         "detected-card-type":"ficon-express-16s-plus",
         "maximum-total-capacity":254,
         "name":"FCP 0100 Z22B-01",
         "object-id":"a44e2648-0a42-11e7-88d2-00106f0d81cb",
         "object-uri":"/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb",
         "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
         "physical-channel-status":"operating",
         "port-count":1,
         "state":"online",
         "status":"active",
         "storage-port-uris":[
            "/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb/storage-ports/0"
         ],
         "type":"fcp",
         "used-capacity":6
      }
   ],
   "available-features-list":[],
   "cpc-properties":{
      "description":"CPC description "
   },
   "hbas":[
      {
         "adapter-port-uri":"/api/adapters/a44e2648-0a42-11e7-88d2-00106f0d81cb/
            storage-ports/0",
         "class":"hba",
         "description":"Systemplatte",
         "device-number":"0150",
         "element-id":"633219d0-5ff6-11e7-92e8-00106f0d81cb",
         "element-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/hbas/
            633219d0-5ff6-11e7-92e8-00106f0d81cb",
         "name":"Bootadapter",
         "parent":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
         "wwpn":"C05076FFE80006A6"
      }
   ],
```


Figure 598. Import DPM Configuration: Request (Part 2)

```
"nics":[
   {
      "class":"nic",
      "description":"Device fuer die freie Welt.",
      "device-number":"0001",
      "element-id":"5d0caee6-5ff4-11e7-90cf-00106f0d81cb",
      "element-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/nics/
         5d0caee6-5ff4-11e7-90cf-00106f0d81cb",
      "mac-address":"12:34:56:78:9a:bc",
      "name":"Netzwerk_Aussen",
      "parent":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
      "ssc-ip-address":null,
      "ssc-ip-address-type":null,
      "ssc-management-nic":false,
      "ssc-mask-prefix":null,
      "type":"iqd",
      "virtual-switch-uri":"/api/virtual-switches/bc6d7ce6-354f-11e7-a1a3-
         00106f0d81cb",
      "vlan-id":null,
      "vlan-type":null
   }
],
"partitions":[
   {
      "acceptable-status":[
         "active"
      ],
      "access-basic-counter-set":false,
      "access-basic-sampling":false,
      "access-coprocessor-group-set":false,
      "access-crypto-activity-counter-set":false,
      "access-diagnostic-sampling":false,
      "access-extended-counter-set":false,
      "access-global-performance-data":false,
      "access-problem-state-counter-set":false,
      "auto-start":false,
      "autogenerate-partition-id":true,
      "boot-configuration-selector":0,
      "boot-device":"test-operating-system",
      "boot-ftp-host":null,
      "boot-ftp-insfile":null,
      "boot-ftp-username":null,
```


Figure 599. Import DPM Configuration: Request (Part 3)

```
        "boot-iso-image-name":null,
        "boot-iso-ins-file":null,
        "boot-logical-unit-number":"0001000000000000",
        "boot-network-device":null,
        "boot-os-specific-parameters":"",
        "boot-record-lba":"",
        "boot-removable-media":null,
        "boot-removable-media-type":null,
        "boot-storage-device":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/
           hbas/633219d0-5ff6-11e7-92e8-00106f0d81cb",
        "boot-timeout":60,
        "boot-world-wide-port-name":"50050763070306A6",
        "class":"partition",
        "cp-absolute-processor-capping":false,
        "cp-absolute-processor-capping-value":1.0,
        "cp-processing-weight-capped":false,
        "cp-processors":6,
        "crypto-configuration":null,
        "current-cp-processing-weight":1,
        "current-ifl-processing-weight":1,
        "degraded-adapters":[],
        "description":"Ihno legt eine LPAR an.",
        "has-unacceptable-status":true,
        "hba-uris":[
           "/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/hbas/633219d0-
              5ff6-11e7-92e8-00106f0d81cb"
        ],
```


Figure 600. Import DPM Configuration: Request (Part 4)

```
            "ifl-absolute-processor-capping":false,
            "ifl-absolute-processor-capping-value":1.0,
            "ifl-processing-weight-capped":false,
            "ifl-processors":0,
            "initial-cp-processing-weight":100,
            "initial-ifl-processing-weight":100,
            "initial-memory":12288,
            "is-locked":false,
            "maximum-cp-processing-weight":999,
            "maximum-ifl-processing-weight":999,
            "maximum-memory":12288,
            "minimum-cp-processing-weight":1,
            "minimum-ifl-processing-weight":1,
            "name":"SUSE_Test",
            "nic-uris":[
               "/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb/nics/5d0caee6-
                  5ff4-11e7-90cf-00106f0d81cb"
            ],
            "object-id":"9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
            "object-uri":"/api/partitions/9d1826c4-5ff3-11e7-b4a6-00106f0d81cb",
            "os-name":"",
            "os-type":"",
            "os-version":"",
            "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
            "partition-id":null,
            "permit-aes-key-import-functions":true,
            "permit-cross-partition-commands":false,
            "permit-des-key-import-functions":true,
            "processor-management-enabled":false,
            "processor-mode":"shared",
            "reserve-resources":false,
            "reserved-memory":0,
            "short-name":"SUSETEST",
            "status":"stopped",
            "threads-per-processor":0,
            "type":"linux",
            "virtual-function-uris":[]
         }
      ],
      "se-version":"2.13.1",
      "virtual-switches":[
         {
            "backing-adapter-uri":"/api/adapters/bc5c79e6-354f-11e7-911e-00106f0d81cb",
            "class":"virtual-switch",
            "description":"",
            "name":"7C0.P0.IQD",
            "object-id":"bc6d7ce6-354f-11e7-a1a3-00106f0d81cb",
            "object-uri":"/api/virtual-switches/bc6d7ce6-354f-11e7-a1a3-00106f0d81cb",
            "parent":"/api/cpcs/f8242e42-c99d-3765-892e-5ddebb74bd2e",
            "port":0,
            "type":"hipersockets"
         }
      ]
   }
```


Figure 601. Import DPM Configuration: Request (Part 5)

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 11 Jun 2018 10:36:04 GMT

<No response body>
```

Figure 602. Import DPM Configuration: Response

Usage notes

- There is an open source command line tool called **zhmc**, available from https://github.com/zhmcclient/zhmccli. This tool can assist with:

– Creating the input data required for this operation by exporting the configuration of an existing machine
– Invoking this operation using that previously generated data on a new machine

Note: When using the tool, ensure to always use the latest version, and to study the corresponding release information and change history.

- By default, only the SERVICE user ID on the HMC has the required task permission for this operation, and that user ID is not enabled for Web Services APIs by default.
- By default, object IDs and element IDs are not preserved; instead, new unique IDs are generated when importing the objects. This prevents the possibility of duplicate IDs, which would occur when both the new and old systems are attached to the same HMC after the configuration is imported with this operation. The **preserve-uris** field can be specified as **true** to preserve the IDs, which can be helpful for situations that depend on the IDs, such as a network (PXE) boot. But care must be taken to never attach both systems to the same HMC if that option is used during this operation. Doing so may result in unpredictable behavior due to the presence of duplicate IDs.
- By default, the WWPNs of the HBAs are not preserved; instead, new WWPNs are generated when importing the HBAs. The generated WWPNs will match the I/O serial number of the new machine. If the new machine will have the same I/O serial number as the old machine, then it is appropriate to specify **preserve-wwpns** as **true** so that the HBAs are imported with their original WWPNs.
- The operation restores property settings in certain adapters (for example: the **type** property for storage adapters). However, it does not restore the **crypto-type** property for crypto adapters; you need to ensure that the target crypto adapters in the new system are set up to match that type. It is recommended that you manually inspect the **crypto-type** property for all crypto adapters prior to import, and make any changes that might be needed, using the **Manage Adapters** task.

List Remote Firmware Updates of a CPC

The `List Remote Firmware Updates of a CPC` operation returns a list of the remote firmware update operations on a CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/remote-firmware-updates
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC object.

Query Parameters

Name	Type	Rqd/Opt	Description
`state`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **state** property. Value must be a valid Remote Firmware Update element object **state** property value.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`remote-firmware-updates`	Array of remote-firmware-update-info objects	A list of the remote firmware update operations scheduled on the CPC. Each element in the list is a remote-firmware-update-info nested object defined in Table 509 on page 1116.

Each nested remote-firmware-update-info object contains the following fields:

Table 509. List Remote Firmware Updates of a CPC: remote-firmware-update-info objects

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path (**element-uri**) of the Remote Firmware Update element object.
`scheduled-execution-time`	Timestamp	The **scheduled-execution-time** property of the Remote Firmware Update element object.
`target-bundle`	String	The **target-bundle** property of the Remote Firmware Update element object.
`state`	String Enum	The **state** property of the Remote Firmware Update element object.

Description

The `List Remote Firmware Updates of a CPC` operations returns a list of the remote firmware update operations that are scheduled to run at a future time on a CPC, or were scheduled and are currently running. The Remote Firmware Update CPC Element URI, scheduled execution start time, bundle level and current state are returned for each.

See "Authorize Remote Firmware Updates" on page 870 for a description of the complete remote firmware update process.

If the **state** query parameter is specified, it is validated to ensure it is a valid value for the Remote Firmware Update CPC Element **state** property. If the value is not valid, a 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those remote firmware updates that have a matching **state** property. If the **state** parameter is omitted, this filtering is not done.

If no remote firmware updates are to be included in the results due to filtering or lack of any remote firmware updates, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-id** is *{cpc-id}*.
- Action/task permission to the **Manage Remote Firmware Updates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1115.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 510. List Remote Firmware Updates of a CPC: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	14	A query parameter defines and invalid value.

Table 510. List Remote Firmware Updates of a CPC: HTTP status and reason codes (continued)		
HTTP error status code	**Reason code**	**Description**
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Remote Firmware Updates** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the Console or the API user does not have object-access permission for it.
	4	This operation is not supported on SE Versions earlier than 2.15.0. [Added by **remote-firmware-update-rc404-4**]

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates HTTP/1.1
x-api-session: 2akxko0yypymzo8mzm2kj4cns5n6mutmppbpdsops6ql2lqn2v
```

Figure 603. List Remote Firmware Updates of a CPC: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:55:11 GMT
content-type: application/json;charset=UTF-8
content-length: 447
{
   "remote-firmware-updates":[
      {
         "element-uri":"/api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates/
           5c7397eb-7401-4db7-86a4-ccab4e4c17d2",
         "scheduled-execution-time":1615270549502,
         "state":"scheduled",
         "target-bundle":"S34"
      },
      {
         "element-uri":"/api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates/
           10f50068-1c4e-488a-8acd-5bfc522222b6",
         "scheduled-execution-time":1615184131054,
         "state":"scheduled",
         "target-bundle":"S33"
      }
   ]
}
```

Figure 604. List Remote Firmware Updates of a CPC: Response

Get CPC Remote Firmware Update Properties

The `Get CPC Remote Firmware Update Properties` operation retrieves the properties of a single Remote Firmware Update element object on a CPC.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/remote-firmware-updates/{remote-firmware-update-id}
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC object, and the URI variable *{remote-firmware-update-id}* is the element ID of the Remote Firmware Update object.

Response body contents

On successful completion, the response body is a JSON object that provides the current values of the properties for the "Remote Firmware Update CPC element object" on page 1032. Field names and data types in the JSON object are the same as the property names and data types defined in the "Data model" on page 1010.

Description

Returns the current values for the properties of the remote firmware update element object as defined in "Remote Firmware Update CPC element object" on page 1032.

See "Authorize Remote Firmware Updates" on page 870 for a description of the complete remote firmware update process.

If the API user does not have action/task permission to the **Manage Remote Firmware Updates** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{cpc-id}* does not identify a CPC object on the Console to which the API user has object-access permission or *{remote-firmware-update-id}* does not identify a Remote Firmware Update element object on the CPC.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-id** is *{cpc-id}*.
- Action/task permission to the **Manage Remote Firmware Updates** task.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1118.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 511. Get CPC Remote Firmware Update Properties: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Manage Remote Firmware Updates** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	This operation is not supported on SE Versions earlier than 2.15.0. [Added by **remote-firmware-update-rc404-4**]
	5	A remote firmware update operation with **element-id** *{remote-firmware-update-id}* does not exist in the CPC.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates/
  10f50068-1c4e-488a-8acd-5bfc522222b6 HTTP/1.1
x-api-session: 1udndd541m1xe0okofnwwtkvl2tz4gcpu89j7i30tj7grdd9ag
```

Figure 605. Get CPC Remote Firmware Update Properties: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:56:00 GMT
content-type: application/json;charset=UTF-8
content-length: 539
{
   "class":"remote-firmware-update",
   "creation-time":1613753551054,
   "element-id":"10f50068-1c4e-488a-8acd-5bfc522222b6",
   "element-uri":"/api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates/
     10f50068-1c4e-488a-8acd-5bfc522222b6",
   "execution-window":60,
   "parent":"/api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c",
   "scheduled-execution-time":1615184131054,
   "service-contact-email-address":"ssrEmail@example.com",
   "service-contact-name":"ssrName",
   "service-contact-telephone-number":"0123456789",
   "state":"scheduled",
   "target-bundle":"S33"
}
```

Figure 606. Get CPC Remote Firmware Update Properties: Response

Delete CPC Remote Firmware Update

The `Delete CPC Remote Firmware Update` operation deletes a remote firmware update operation scheduled to run at a future time on the CPC.

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/remote-firmware-updates/{remote-firmware-update-id}
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC on which the remote firmware update operation to delete resides, and the URI variable *{remote-firmware-update-id}* is the element ID of the remote firmware update operation.

Description

The `Delete CPC Remote Firmware Update` operation deletes a remote firmware update operation scheduled to run at a future time on a CPC.

See “Authorize Remote Firmware Updates” on page 870 for a description of the complete remote firmware update process.

If the API user does not have action/task permission to the **Cancel Scheduled Update** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{cpc-id}* does not identify a CPC object on the Console to which the API user has object-access permission or *{remote-firmware-update-id}* does not identify a Remote Firmware Update element object on the CPC. A 409 (Conflict) status code is returned if the target CPC is busy, if the remote firmware update operation identified by **remote-firmware-update-id** is already running, or if the operation has already completed. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC. [Updated by feature **rcl-progress**]

If the request is valid, the identified remote firmware update is deleted from the CPC.

Authorization requirements

This operation has the following authorization requirement:

- Object-access permission to the CPC whose **object-id** is *{cpc-id}*.
- Action/task permission to the **Cancel Scheduled Update** task.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 512. Delete CPC Remote Firmware Update: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Cancel Scheduled Update** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the HMC or the API user does not have object-access permission for it.
	4	This operation is not supported on SE Versions earlier than 2.15.0. [Added by **remote-firmware-update-rc404-4**]
	5	A remote firmware update operation with **element-id** *{remote-firmware-update-id}* does not exist in the CPC with the **object-id** *{cpc-id}*.
409 (Conflict)	2	The CPC object with the **object-id** *{cpc-id}* was busy and the request timed out.
	342	The value of the **state** property of the remote firmware update operation with **element-id** *{remote-firmware-update-id}* is **"running"** and it can therefore no longer be deleted.
	382	The value of the **state** property of the remote firmware update with **element-id** *{remote-firmware-update-id}* indicates that the operation has already completed and it can therefore no longer be deleted. [Added by feature **rcl-progress**]
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/remote-firmware-updates/
  10f50068-1c4e-488a-8acd-5bfc522222b6 HTTP/1.1
x-api-session: 4npzf2i29b0euw98nguczjmv7sy7gg9jkxrbn2q8hf9k7uhp38
```

Figure 607. Delete CPC Remote Firmware Update: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Fri, 19 Feb 2021 17:56:35 GMT

<No response body>
```

Figure 608. Delete CPC Remote Firmware Update: Response

Get Logical Partition Resource Assignments

The `Get Logical Partition Resource Assignments` operation retrieves logical partition processor allocation information for the CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-lpar-resource-assignments
```

In this request, the URI variable *{cpc-id}* is the object ID of a CPC object for which logical partition resource assignment information is to be retrieved.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`lpar-resource-assignments-info`	String	A JSON string that describes logical partition allocation information for the CPC object. The exact format of this information should be used under the direction of product support and is subject to change at any time.

Description

This operation retrieves information for the allocation of processors to logical partitions. Use of this information should be done under the direction of product support.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/Task permission to the **View Partition Resource Assignments** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1121.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 513. Get Logical Partition Resource Assignments: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Get LPAR Controls

The `Get LPAR Controls` operation retrieves all the LPAR controls data for each logical partition of a CPC object. This is the same LPAR controls data that can be individually queried for a Logical Partition object using the `Get Logical Partition Properties` operation. This operation is supported using the BCPii interface. For the web services interface, the requested information is only returned for Logical Partition objects to which the API user has object-access permission. For the BCPii interface, the requested information is only returned for Logical Partition objects to which the source partition has receive BCPii security permission.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-lpar-controls
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a LPAR controls related property name defined in the Logical Partition object's data model. **Note:** The **object-uri**, **object-id**, and **name** fields of each returned **lpar-controls-info** will always be included in the response body, regardless of whether they are included in the properties filter or not.
`cached-acceptable`	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**. The default is **false**.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Name	Type	Description
`lpar-controls`	Array of lpar-controls-info objects	Array of nested lpar-controls-info objects as described in the next table.

Each nested lpar-controls-info object contains the following fields:

Name	Type	Description
`object-uri`	String/URI	The canonical URI path of the Logical Partition object, of the form `/api/logical-partitions/`*`{logical-partition-id}`* where *{logical-partition-id}* is the value of the **object-id** property of the Logical Partition object.
`object-id`	String (36)	The object identifier for the Logical Partition object.
`name`	String (1-8)	The name of the logical partition.
`initial-processing-weight`	Integer (1-999)	The relative amount of shared general purpose processor resources allocated to the logical partition.
`initial-processing-weight-capped`	Boolean	Whether the initial processing weight for general purpose processors is a limit or a target. **True** Indicates that the initial general purpose processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources. **False** Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.

Name	Type	Description
`minimum-processing-weight`	Integer	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be less than or equal to the initial-processing-weight property.
`maximum-processing-weight`	Integer	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Defines the maximum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be greater than or equal to the initial-processing-weight property.
`current-processing-weight`	Integer (1-999)	The relative amount of shared general purpose processor resources currently allocated to the logical partition.

Name	Type	Description
`current-processing-weight-capped`	Boolean	Whether the current general purpose processing weight is a limit or a target. **True** Indicates that the current general purpose processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current general purpose processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
`workload-manager-enabled`	Boolean	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. **True** Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. **False** Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition.
`absolute-processing-capping`	absolute-capping object	The amount of absolute capping applied to the general purpose processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
`defined-capacity`	Integer	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed to by z/OS Workload Manager for the purpose of software pricing. **0** No defined capacity is specified for this logical partition. **1-nnnn** Represents the amount of defined capacity specified for this logical partition.
`initial-ifl-processing-weight`	Integer (1-999)	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.

Name	Type	Description
`initial-ifl-processing-weight-capped`	Boolean	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. **True** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources **False** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.
`minimum-ifl-processing-weight`	Integer	The minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.

Name	Type	Description
`maximum-ifl-processing-weight`	Integer	The maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
`current-ifl-processing-weight`	Integer (1-999)	The current relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
`current-ifl-processing-weight-capped`	Boolean	Whether the current Integrated Facility for Linux (IFL) processing weight is a limit or a target. **True** Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
`absolute-ifl-capping`	absolute-capping object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
`initial-ziip-processing-weight`	Integer (1-999)	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.

Name	Type	Description
`initial-ziip-processing-weight-capped`	Boolean	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. **True** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. **False** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.
`minimum-ziip-processing-weight`	Integer	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.

Name	Type	Description
`maximum-ziip-processing-weight`	Integer	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
`current-ziip-processing-weight`	Integer (1-999)	The current relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
`current-ziip-processing-weight-capped`	Boolean	Whether the current z Integrated Information Processors (zIIP) processing weight is a limit or a target. **True** Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
`absolute-ziip-capping`	absolute-capping object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
`initial-cf-processing-weight`	Integer (1-999)	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.

Name	Type	Description
`initial-cf-processing-weight-capped`	Boolean	Indicates whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. **True** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. **False** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition is not capped. It represents the share of Internal Coupling Facility (ICF) processor resources guaranteed to a logical partition when all Internal Coupling Facility (ICF) processor resources are in use. Otherwise, when excess Internal Coupling Facility (ICF) processor resources are available, the logical partition can use them if necessary.
`minimum-cf-processing-weight`	Integer	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: **1-999** The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
`maximum-cf-processing-weight`	Integer	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.

Name	Type	Description
`current-cf-processing-weight`	Integer (1-999)	The current relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
`current-cf-processing-weight-capped`	Boolean	Indicates whether the current Internal Coupling Facility (ICF) processing weight is a limit or a target. **True** Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
`absolute-cf-capping`	absolute-capping object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.

Note: All of the fields in the lpar-controls-info object are writable with the exception of the **object-uri**, **object-id**, and **name** fields.

Description

This operation returns LPAR controls information for Logical Partitions associated with the specified CPC object, filtered according to the query parameters and the permissions of the API user.

Some of this object's property values are periodically fetched from the Support Element and cached for quick access by the APIs. Due to the nature of this caching support, the cached value of a property may differ from the actual value at any point in time. While the cache is kept reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, object-access permission for each Logical Partition to be included in the results
- For the BCPii interface the source partition must have receive BCPii security controls permissions for each Logical Partition to be included in the results.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1123.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/1803ceb9-928f-3d68-8501-d0054851afcf/operations/
  get-lpar-controls HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 609. Get LPAR Controls: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Wed, 10 Aug 2021 08:18:00 GMT
content-type: application/json;charset=UTF-8
content-length: 3586
{
   "lpar-controls": [
      {
         "absolute-cf-capping": {
            "type": "none"
         },
         "absolute-ifl-capping": {
            "type": "none"
         },
         "absolute-processing-capping": {
            "type": "none"
         },
         "absolute-ziip-capping": {
            "type": "none"
         },
         "current-cf-processing-weight": null,
         "current-cf-processing-weight-capped": null,
         "current-ifl-processing-weight": null,
         "current-ifl-processing-weight-capped": null,
         "current-processing-weight": 10,
         "current-processing-weight-capped": false,
         "current-ziip-processing-weight": null,
         "current-ziip-processing-weight-capped": null,
         "defined-capacity": 0,
         "initial-cf-processing-weight": null,
         "initial-cf-processing-weight-capped": null,
         "initial-ifl-processing-weight": null,
         "initial-ifl-processing-weight-capped": null,
         "initial-processing-weight": 10,
         "initial-processing-weight-capped": false,
         "initial-ziip-processing-weight": null,
         "initial-ziip-processing-weight-capped": null,
         "maximum-cf-processing-weight": null,
         "maximum-ifl-processing-weight": null,
         "maximum-processing-weight": 0,
         "maximum-ziip-processing-weight": null,
         "minimum-cf-processing-weight": null,
         "minimum-ifl-processing-weight": null,
         "minimum-processing-weight": 0,
         "minimum-ziip-processing-weight": null,
         "name": "LP01",
         "object-id": "e953b32a-722c-3d32-820b-0eec1927bb0c",
         "object-uri": "/api/logical-partitions/e953b32a-722c-3d32-820b-0eec1927bb0c",
         "workload-manager-enabled": false
      },
      {
         "absolute-cf-capping": {
            "type": "none"
         },
         "absolute-ifl-capping": {
            "type": "none"
         },
         "absolute-processing-capping": {
            "type": "none"
         },
         "absolute-ziip-capping": {
            "type": "none"
         },
```

Figure 610. Get LPAR Controls: Response (part 1)

```
            "current-cf-processing-weight": null,
            "current-cf-processing-weight-capped": null,
            "current-ifl-processing-weight": null,
            "current-ifl-processing-weight-capped": null,
            "current-processing-weight": 10,
            "current-processing-weight-capped": false,
            "current-ziip-processing-weight": null,
            "current-ziip-processing-weight-capped": null,
            "defined-capacity": 0,
            "initial-cf-processing-weight": null,
            "initial-cf-processing-weight-capped": null,
            "initial-ifl-processing-weight": null,
            "initial-ifl-processing-weight-capped": null,
            "initial-processing-weight": 10,
            "initial-processing-weight-capped": false,
            "initial-ziip-processing-weight": null,
            "initial-ziip-processing-weight-capped": null,
            "maximum-cf-processing-weight": null,
            "maximum-ifl-processing-weight": null,
            "maximum-processing-weight": 0,
            "maximum-ziip-processing-weight": null,
            "minimum-cf-processing-weight": null,
            "minimum-ifl-processing-weight": null,
            "minimum-processing-weight": 0,
            "minimum-ziip-processing-weight": null,
            "name": "LP02",
            "object-id": "bfc90b55-9a12-3132-a0ca-bcb78540e294",
            "object-uri": "/api/logical-partitions/bfc90b55-9a12-3132-a0ca-bcb78540e294",
            "workload-manager-enabled": false
        }
    ]
}
```


Figure 611. Get LPAR Controls: Response (part 2)

Update LPAR Controls

The `Update LPAR Controls` operation updates one or more of the writable values returned from the `Get LPAR Controls` operation. This is the same LPAR controls data that can be individually updated for a Logical Partition object using the `Update Logical Partition Properties` operation. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/update-lpar-controls
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain an array of lpar-controls-info objects that contains the fields and new values that are to be updated. The request body can and should omit fields that are not to be changed by this operation. Fields for which no input value is provided remain unchanged by this operation. Each array element must contain at least one of the **object-uri**, **object-id**, or **name** fields. If more than one of these fields is specified, then they must be consistent, or the operation will be rejected.

Description

The request body object is validated against the lpar-controls-info object definition and the current set of Logical Partition objects for the CPC to ensure that the request body contains only writable properties and the data types of those properties are as required. Object access permission for the associated Logical Partition object is required for any lpar-control-info object specified in the request body. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface:
  - Action/task permission for the **Image Details** task.
  - If the **workload-manager-enabled**, **defined-capacity**, **absolute-processing-capping**, **absolute-ifl-capping**, **absolute-ziip-capping**, or **absolute-cf-capping** properties are being updated then object-access permission to the CPC object designated by *{cpc-id}*.
  - Object-access permission for any Logical Partition object that has an associated lpar-controls-info object in the request.
- For the BCPii interface:
  - If the **workload-manager-enabled**, **defined-capacity**, **absolute-processing-capping**, **absolute-ifl-capping**, **absolute-ziip-capping**, or **absolute-cf-capping** properties are being updated then receive BCPii permission for the CPC object designated by *{cpc-id}*.
  - Receive BCPii security permission for any Logical Partition object that has an associated lpar-controls-info object in the request.

HTTP status and reason codes

On success, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation. It is important to note that all the updates are performed as a single transaction, which means that either all changes are made (success) or no changes are made (failure).

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 514. Update LPAR Controls: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes. **Note:** This error status is also used for the case when an inconsistent set of **object-id**, **object-uri**, or **name** is specified.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The **object-id**, **object-uri**, or **name** in the request body does not designate an existing Logical Partition object.
409 (Conflict	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	0	A general conflict was detected during common request validation. See the returned message field for details.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/1803ceb9-928f-3d68-8501-d0054851afcf/operations/
  update-lpar-controls HTTP/1.1
x-api-session: 440mepm56w27o7965y61kbkm4g422hja984lrm74mc2wdpf3vw
content-type: application/json
content-length: 1279
{
   "lpar-controls": [
      {
         "absolute-processing-capping": {
            "type": "none"
         },
         "current-processing-weight": 110,
         "current-processing-weight-capped": false,
         "defined-capacity": 0,
         "initial-processing-weight": 120,
         "initial-processing-weight-capped": false,
         "maximum-processing-weight": 200,
         "minimum-processing-weight": 10,
         "name": "LP01",
         "object-id": "e953b32a-722c-3d32-820b-0eec1927bb0c",
         "object-uri": "/api/logical-partitions/e953b32a-722c-3d32-820b-0eec1927bb0c",
         "workload-manager-enabled": true
      },
      {
         "absolute-ifl-capping": {
            "type": "none"
         },
         "current-ifl-processing-weight": 50,
         "current-ifl-processing-weight-capped": false,
         "initial-ifl-processing-weight": 75,
         "initial-ifl-processing-weight-capped": false,
         "maximum-ifl-processing-weight": 100,
         "minimum-ifl-processing-weight": 25,
         "name": "LP02",
         "object-id": "bfc90b55-9a12-3132-a0ca-bcb78540e294",
         "object-uri": "/api/logical-partitions/bfc90b55-9a12-3132-a0ca-bcb78540e294",
         "workload-manager-enabled": false
      }
   ]
}
```

Figure 612. Update LPAR Controls: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Tue, 10 Aug 2021 08:01:20 GMT
<No response body>
```

Figure 613. Update LPAR Controls: Response

CPC Single Step Install

The `CPC Single Step Install` operation asynchronously performs a backup of a single CPC's firmware and then retrieves, installs, and activates a new bundle of firmware.

HTTP method and URI

`POST /api/cpcs/{cpc-id}/operations/single-step-install`

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`bundle-level`	String	Optional	Name of the bundle to be installed
`accept-firmware`	Boolean	Optional	Accept the previous bundle level before installing the new level. Default value: **true**
`ftp-retrieve`	Boolean	Optional	Retrieve internal code changes from an FTP server. Default value: **false** [Added by feature **cpc-delete-retrieved-internal-code**]
`ftp-server-host`	String	Optional	The hostname for the FTP server. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **cpc-delete-retrieved-internal-code**]
`ftp-server-user`	String	Optional	The username for FTP server login. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **cpc-delete-retrieved-internal-code**]
`ftp-server-password`	String	Optional	The password for FTP server login. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **cpc-delete-retrieved-internal-code**]
`ftp-server-directory`	String	Optional	The directory to access on the FTP server. **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **cpc-delete-retrieved-internal-code**]
`ftp-server-protocol`	String Enum	Optional	The protocol used to connect to the FTP server. Valid values: • **"ftp"** • **"ftps"** • **"sftp"** **Note:** This field is required if **ftp-retrieve** is **true**. [Added by feature **cpc-delete-retrieved-internal-code**]

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates or used to request cancellation of the operation.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in the "Job status and reason codes" on page 1140. The **job-results** field is **null** when this operation is successful. When it is not successful or partially successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful, or a description of firmware updates that are still pending when the operation was partially successful.

Description

The `CPC Single Step Install` operation installs a firmware bundle on a CPC. The **ec-mcl-description** property of the target CPC object provides information about the firmware levels that are known to the CPC.

Note that it is not possible to remove firmware updates with this operation. Specifying a **bundle-level** that targets firmware updates that are already installed will result in an error rather than a removal of the firmware down to the specified level.

Note that it is not possible to view MCL alerts with this operation. Specifying a **bundle-level** that targets firmware updates that trigger an alert during install and activate will result in an error rather than a display of the alert. If the user wishes to proceed with the operation, they will have to log on to the HMC that manages the target CPC via the local graphical user interface (GUI) or a remote web browser, then navigate to the **Single Step Internal Code Changes** task and rerun the operation.

The internal code that is installed during this operation can be retrieved from either the remote support system or a specified FTP server. In either case, an attempt will be made to retrieve all available internal code changes. If the **bundle-level** field is specified, then all retrieved internal code changes up to the specified bundle boundary will be installed on the target CPC. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed.

If the **ftp-retrieve** field is specified as **false**, then all possible internal code changes will be retrieved from the remote support system. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed on the target CPC. Otherwise, the retrieved internal code changes will be installed up to the specified bundle boundary.

If the **ftp-retrieve** field is specified as **true**, then all possible internal code changes will be retrieved from an FTP server. In order to achieve this, the **ftp-server-host**, **ftp-server-user**, **ftp-server-password**, **ftp-server-directory**, and **ftp-server-protocol** fields must all be included in the initial request. If the retrieval was successful, then the set of internal code changes that will be installed depends on the **bundle-level** field. If the **bundle-level** field is omitted, then all retrieved internal code changes will be installed on the target CPC. Otherwise, the retrieved internal code changes will be installed up to the specified bundle boundary. If an error occurs trying to locate the server with the specified hostname, or if authentication to the server cannot be established with provided user credentials, or if the internal code changes cannot be retrieved from their specified location, then a 409 (Conflict) will be returned.

If the API user does not have action/task permission to the **Single Step Internal Code Changes** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{cpc-id}* does not identify a CPC object on the Console to which the API user has object-access permission. A 409 (Conflict) status code is returned if the target CPC is busy on the Console. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, a 202 (Accepted) response is returned and an asynchronous job is started to install the firmware updates identified by the **bundle-level** request body field. Once started, the asynchronous job performs additional validation of the request body fields. If the **bundle-level** does not exist or is known but not yet retrieved or is already installed on the target CPC, or if the targeted set of firmware contains disruptive updates, a 400 (Bad Request) is returned in response to a Query Job Status Request. A 409 (Conflict) is returned if Change Management is not enabled, or if a connection to the Support System is not available on the target CPC.

If the request body contents are valid, the firmware identified by the **bundle-level** request body field is installed. The install process includes the following steps:

- A backup of the target CPC is performed to its Support Element hard drive.
- If the value of the **accept-firmware** is **true**, the firmware currently installed on the target CPC is accepted. Note that once firmware is accepted, it cannot be removed.
- The uninstalled firmware identified by the **bundle-level** field is installed.
- The newly installed firmware is activated, which includes rebooting the CPC's Support Element.

If an error occurred when installing updates, any updates that were successfully installed are rolled back. If a failure occurs after the firmware is accepted, the firmware remains accepted.

When the asynchronous job competes, the response to a `Query Job Status` request will include a **status** of **"complete"**. If the operation was successful, the completion status will be 204 (No Content). It is possible that additional actions, such as configuring virtual adapters off and on, may be required to fully activate certain firmware updates. If this is the case, the asynchronous job completion status will be 200 (OK), indicating overall success of the operation, but the result will contain a **job-results** field with a message indicating some firmware updates remain in a pending state. The **View Internal Code Changes Summary** task on the Console or Support Element user interface will provide a list of the additional actions that are required. It is not possible to query this information via the API.

This operation supports cancellation of its asynchronous processing identified by the `Job URI` provided in the response body. Use the `Cancel Job` operation to request cancellation. Note there are only a few interruption points in the firmware install process, so it may be some time before the job is canceled, and after some point, will continue to completion. The job status and reason codes will indicate whether the job was canceled or ran to completion. If the job is successfully canceled, any steps that were successfully completed will not be rolled back.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object whose object ID is *{cpc-id}*
- Action/task permission for the **Single Step Internal Code Changes** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1137.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	1	The API user does not have action/task permission to the **Single Step Internal Code Changes** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the Console or the API user does not have object-access permission to the object.
409 (Conflict)	2	The CPC object with the **object-id** *{cpc-id}* was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the Console is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
200 (OK)	N/A	The operation completed successfully, but some firmware remains in a pending state.
204 (No Content)	N/A	The operation completed successfully.
400 (Bad Request)	354	The bundle identified by **bundle-level** does not exist on the target CPC.
	355	The bundle identified by **bundle-level** is known on the target CPC, but has not yet been retrieved.
	356	The bundle identified by **bundle-level** is already installed on the target CPC.
	357	There are MCLs currently installed on the target CPC that are beyond the bundle identified by **bundle-level**.
	358	The bundle identified by **bundle-level** has been marked as invalid for install.
	359	The targeted set of firmware updates contains one or more updates that are disruptive to CPC operations.
	401	There are MCL alerts present for install and activate of the bundle identified by **bundle-level**. Log on to the GUI of the Console managing the target CPC, and then navigate to the **Single Step Internal Code Change** task and rerun the operation.

Job status code	Job reason code	Description
409 (Conflict)	341	The CPC identified by *{cpc-id}* does not have an active connection to the Support System.
	342	Connection to the FTP server specified in the **ftp-server-host** field failed. Verify that the hostname is well-defined and exists.
	343	Authentication to the FTP server failed with the provided values for the **ftp-server-user** and **ftp-server-password** fields.
	344	No host key can be found while attempting to access the FTP server specified in the **ftp-server-host** field via the SFTP protocol. Navigate to the **Manage SSH Keys** task in order to add a key for the host.
	345	Connection to the FTP server specified in the **ftp-server-host** field using the FTPS protocol could not be established due to the host's certificate not being recognized. Navigate to the **Certificate Management** task to import the certificate.
	347	Failure occurred while attempting to retrieve the internal code changes contained in the **ftp-server-directory** field on the FTP server.
	367	Change Management is not enabled on the CPE identified by *{cpc-id}*.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/4424b653-909a-345e-ad68-bc1da45c446c/operations/single-step-install HTTP/1.1
x-api-session: 1lhmy9pgq9lwimszhhg1ta210lo4t3xmk8dch1gzp5wbm06ujz
Content-Type: application/json
Content-Length: 49
{
   "accept-firmware":false,
   "bundle-level":"S01"
}
```

Figure 614. CPC Single Step Install: Request

```
202
Content-Type: application/json;charset=UTF-8
Content-Length: 60
{
   "job-uri":"/api/jobs/6b8834b4-c259-11ec-a05d-fa163e920859"
}
```

Figure 615. CPC Single Step Install: Response

Import Secure Execution Key

The `Import Secure Execution Key` operation imports the global Hyper Protect key, host key, or host import key for secure execution to the system. This operation is supported using the BCPii interface. [Added by feature **secure-execution-key-management**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-se-key
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`file-content`	String	Required	Base64-encoded string representation of the entire content of the global Hyper Protect key bundle file, the host key bundle file, or the host import key bundle file to be imported to the system.
`type`	String Enum	Required	The type of the key bundle file content. One of: • **"global"** - global Hyper Protect key • **"host"** - host key • **"host-import"** - host import key
`force`	Boolean	Optional	Whether the installation should proceed (**true**) or not (**false**) when certain kinds of verification of the key bundle file fail. Default value: **false** **Note:** Not all kinds of verification errors can be ignored by the system. Specifying true does not guarantee the request will not be rejected due to verification failure(s).

Description

This operation decodes the **file-content** using the Base64-encoded format described in RFC 4648, verifies the content of the file, and then installs the corresponding key bundle on the system. Once the key bundle has been installed successfully, the corresponding new key will become the primary key and the previous key will become the secondary key.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and the SE version must be at 2.16.0 with the suitable MCL bundle, or a later SE version. If any of these conditions is not met, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have action/task permission to the **Manage Secure Execution Keys** and the **Import Secure Execution Keys** tasks; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned. If the key bundle file type specified in the **type** field of the request body does not match the type extracted from the file content, a 400 (Bad Request) status code is returned. If there is any error detected from the decoding of the file content, certain errors that cannot be ignored are detected while verifying the bundle file content, or certain errors that can be ignored are detected but the **force** field in the request body is **false,** a 400 (Bad Request) status code is returned. If the target CPC is not power-on reset complete, the primary and alternate SEs are not communicating, or the system is busy processing other operations, a 409 (Conflict) status code is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Manage Secure Execution Keys** task.
  - Action/task permission for the **Import Secure Execution Keys** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	378	Decoding of the Base64-encoded key bundle file content failed.
	379	Verification of the key bundle file content failed. The **error-details** field of the response body contains an array of se-key-error-info objects indicating one or more errors have occurred. The se-key-error-info object is described in the next table.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	2	The system is busy processing other operations.
	8	The target system is not power-on reset complete, or the primary and alternate SEs are not communicating.
500 (Server Error)	0	Unexpected error occurred when processing the Web Services API operation.
	380	Unexpected error occurred during the installation of the key bundle file. The **error-details** field of the response body contains an array of se-key-error-info objects indicating one or more errors have occurred. The se-key-error-info object is described in the next table.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

The se-key-error-info object contains the following fields:

Table 515.

Field name	Type	Description
error-code	Integer	The error code
message	String	The error message

The possible error codes with messages for the se-key-error-info object are:

Table 516.

error-code	Description
0	A general error occurred during Secure Execution key management.
9	An error occurred during bundle verification; invalid bundle type
12	An error occurred during bundle verification; bundle timestamp is older than current primary bundle.
42	An error occurred while attempting to install bundle; temp bundle could not be found.
44	An error occurred while attempting to install bundle; unsupported bundle type.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/e5ae3ab6-ac8d-33bc-9739-eb142d89804d/operations/import-se-key HTTP/1.1
x-api-session: 2kk848szmu8mo00lkj4c19254fiejeanyv316j0d5d4uppgp8t
content-type: application/json
content-length: 131
{
     “file-content”: “ejfieurn3kridoru3jrndoapsoek3jdoei3jenfoapsoeifj3ndofien342jdoej”,
     “type”: “global”,
     “force”: false
}
```

Figure 616. Import Secure Execution Key: Request

```
204 No content
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Fri, 20 Jan 2023 13:46:12 GMT

<No response body>
```

Figure 617. Import Secure Execution Key: Response

Delete Secure Execution Key

The `Delete Secure Execution Key` operation deletes a secondary global Hyper Protect key or a secondary host key for secure execution from the system. This operation is supported using the BCPii interface. [Added by feature **secure-execution-key-management**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/delete-se-key
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`type`	String Enum	Required	The type of the secondary key to be deleted from the system. One of: • **"global"** - global Hyper Protect key • **"host"** - host key

Description

This operation deletes the secondary key from the system based on the key type specified in the **type** field of the request body.

The URI path must designate an existing CPC object, the API user must have object-access permission to it, and the SE version must be at 2.16.0 with the suitable MCL bundle, or a later SE version. If any of these conditions is not met, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have action/task permission to the **Manage Secure Execution Keys** and the **Delete Secondary Secure Execution Key** tasks; otherwise, status code 403 (Forbidden) is returned. For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object; otherwise, status code 403 (Forbidden) is returned. If the target CPC is not power-on reset complete, the primary and alternate SEs are not communicating, or the system is busy processing other operations, a 409 (Conflict) status code is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Manage Secure Execution Keys** task.
  - Action/task permission for the **Delete Secondary Secure Execution Key** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The CPC designated by the request URI does not support this operation.
409 (Conflict)	2	The system is busy processing other operations.
	8	The target system is not power-on reset complete, or the primary and alternate SEs are not communicating.
	381	The specified key type is not installed on the system.
500 (Server Error)	0	Unexpected error occurred when processing the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/e5ae3ab6-ac8d-33bc-9739-eb142d89804d/operations/delete-se-key HTTP/1.1
x-api-session: 2kk848szmu8mo00lkj4c19254fiejeanyv316j0d5d4uppgp8t
content-type: application/json
content-length: 23
{
     “type”: “global”
}
```

Figure 618. Delete Secure Execution Key: Request

```
204 No content
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Fri, 20 Jan 2023 13:46:12 GMT

<No response body>
```

Figure 619. Delete Secure Execution Key: Response

Import CPC Certificate

The `Import CPC Certificate` operation imports a certificate based on its type to a CPC. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/import-certificate
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`name`	String	Required	The value to be set as the certificate's **name** property.
`description`	String	Optional	The value to be set as the certificate's **description** property.
`certificate`	String	Required	The Base64-encoded string form of the CPC certificate to import.
`type`	String Enum	Required	The value to be set as the certificate's **type** property.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`certificate-uri`	String/ URI	The URI of the newly created Certificate object.

Description

This operation imports a certificate based on its type to a CPC. An Inventory Change notification is emitted asynchronously to this operation.

If the Certificate being imported has the same name as an existing certificate or there was a problem with the certificate or the file being imported contained multiple certificates, a 400 (Bad Request) status code is returned. A 404 (Not Found) status code is returned if the request URI does not designate an existing CPC, or if the API user does not have object-access permission to the object. If the API user does not have action/task permission to Import Secure Boot Certificates task, 403 (Forbidden) status code is returned. If attempting to import a certificate to an unmanaged CPC, or if importing the Certificate would exceed the Certificate limit of 100 per CPC, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Import Secure Boot Certificates** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1147

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The value of a field does not provide a unique value for the corresponding data model property as required.
	368	There was a problem with the certificate. This could be due to bad formatting, not being able to decode the certificate, etc.
	369	The operation cannot be completed because the certificate string being imported contains multiple certificates. Only one certificate can be imported at a time.
	381	The operation cannot be completed because the certificate is expired.
403 (Forbidden)	0	The request used the BCPii interface and the source CPC does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	329	The operation cannot be performed because the CPC identified by the request URI is an unmanaged CPC, which is not supported by this operation.
	371	The operation could not be performed because importing this certificate would exceed the limit of 100 certificates per CPC.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4/operations/import-certificate HTTP/1.1
x-api-session: 67fbo5w4o1wwpkv2juhbrpux5k0rc5cbmt9594r6fxkl0v5xtv
Content-Type: application/json
Content-Length: 1385
{
   "certificate":"MIIDtDCCApygAwIBAgIUZItvCP3Qvs4XyplEsXl/
g344spAwDQYJKoZIhvcNAQELBQAwWjEpMCcGA1UEAwwgTGludXggU2VjdXJlIEJvb3QgKGJyaW5nLXVwIGtleSkxLTArBgkq
hkiG9w0BCQEWHnBldGVyLm9iZXJwYXJsZWl0ZXJAZGUuaWJtLmNvbTAeFw0yMjAyMTExNzQ3NDBaFw0zMTExMTExNzQ3NDBa
MFoxKTAnBgNVBAMMIExpbnV4IFNlY3VyZSBCb290IChicmluZy11cCBrZXkpMS0wKwYJKoZIhvcNAQkBFh5wZXRlci5vYmVy
cGFybGVpdGVyQGRlLmlibS5jb20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCs3IVBAHIBQHCm4KL/
8tIyMpOMWTQi9lsSz4J/OXGfuW+IwKZnUpxCoGM+3xi6qXvTf5jgZxvLCSxM4Qut303l/
4nG7pyNBQ8IFQBwmeibV04HkVFlHVZVPawanKJlSfIZyq7a9VOqHtWD1go81UchVOnjzSrj4fMHtB0AuidjUAQYY5HXSVEje
EFgv+af1urg1VvhWNGnYPkAzqFQhPgki/EA1ydlQLfTAWgDlnvfgLaDVRnf83vo7PpRpY5gsRUu/
eTF5CkrKO2+QUc5OsgaHQXJ7Hv51ad10JEYyHELSuoCVQxD0dfwf8XMqbs4DO1lvDY6HGa6xf8+cmT0n8g3AgMBAAGjcjBwM
B0GA1UdDgQWBBQURrE+E08OW/buPUlnIQvpPi3aaDAfBgNVHSMEGDAWgBQURrE+E08OW/
buPUlnIQvpPi3aaDAMBgNVHRMBAf8EAjAAMAsGA1UdDwQEAwIHgDATBgNVHSUEDDAKBggrBgEFBQcDAzANBgkqhkiG9w0BAQ
sFAAOCAQEAW+XXxWd7dQU68YapkzNY9XiGOCJdHBZ9yrNBjAqi5KfG4ASjyZE67fmdqo5fzf3SFX0kIMx/
9FUz3CsQbDBK7eMZO3ZagelLTR3BkiiCi9tzTLEVrOFkFB3m9Vz7YbPLssjoGMLo7VagMIZSRUDb+XDA8NJemaYLocPShNxB
wCtEdU6yn5Rc3GxkVHKaljubLV93QwvqF+ObMv3QAmKB/
mci9z+ZObZWN8MgCLXJebRQAPlmvaxS+y0vXNd7getApyfXWWyd63PalU/
Ie8aGXNsdJj0MAQu8Ku1Fd8BRDdyqQyYxKSTJCYV108aw29DhRSBOGLAS2tOK/BJ5S2r70w==",
   "description":"Certificate for Linux",
   "name":"Linux certificate",
   "type":"secure-boot"
}
```

Figure 620. Import CPC Certificate: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 19:50:55 GMT
Content-Type: application/json;charset=UTF-8
Content-Length: 76
{
   "certificate-uri":"/api/certificates/dab30826-48d4-11ed-87c1-fa163e6f7e7e"
}
```

Figure 621. Import CPC Certificate: Response

Report a CPC Problem

The `Report a CPC Problem` operation reports and requests service for a problem on a CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface. [Added by feature **report-a-problem**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/report-problem
```

In this request, the URI variable *{cpc-id}* is the Object ID of the CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`customer-name`	String (0-50)	Optional	Name of the customer. May not contain any double-byte characters or ";". Default: "Unknown"

Name	Type	Req/Opt	Description
`customer-phone-number`	String (0-20)	Optional	Phone number of customer. May not contain any double-byte characters or ";". Default: "Unknown"
`problem-description`	String (1-510)	Required	Description of the problem. May not contain any double-byte characters or ";".
`problem-type`	String Enum	Required	Identifies the type of problem. One of: • **"power"** - Report a problem with the power subsystem. • **"cpc"** - Report a problem with hardware in the processor subsystem. • **"lan"** - Report a problem with the local area network (LAN). • **"software"** - Report a problem with an operating system or other software. • **"io"** -Report a problem with hardware in the input/output (I/O) configuration. • **"health"** - Report the state of the system before applying a maintenance action. • **"other"** - Report a problem that is not adequately described by any other problem type. • **"test"** - Test whether problems can be reported for the selected system.

Description

The `Report a CPC Problem` operation reports a problem for a CPC object and requests service to repair it.

Problems are reported to the support system for the provided system. Reporting a problem sends the information provided in the request and the machine information that identifies the system to the service provider.

Automatic service call reporting must be enabled on the SE associated with the CPC object via the **Remote Service** task to use this operation. If the SE associated with the CPC object does not have automatic service call reporting enabled, a 409 (Conflict) status code is returned.

Upon successful problem creation, a 204 (No Content) status code is returned. If the API user does not have action/task permission to the **Report a Problem** task, a 403 (Forbidden) status code is returned. If the SE associated with the CPC object is unreachable, a 503 (Service Unavailable) status code is returned. The URI path must designate an existing CPC and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Action/task permission for the **Report A Problem** task.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object
	1	The API user does not have the required action/task permissions.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	600	The operation cannot be performed because the SE does not have automatic service call reporting enabled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/0b9e5710-4edc-3402-adf8-809f4d445242/operations/report-problem HTTP/1.1
x-api-session: 2rbq3ademcduira0f4jfawsncvynec6lxlmdgxakzwqui9u6up
Content-Type: application/json
Content-Length: 142
{
  "customer-name":"Tester",
  "customer-phone-number":"888-888-8888",
  "problem-description":"This is a test IO problem",
  "problem-type":"io"
}
```

Figure 622. Report a CPC Problem: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 06 Feb 2023 23:25:25 GMT
<No response body>
```

Figure 623. Report a CPC Problem: Response

Get CPC Historical Sustainability Data

Use the `Get CPC Historical Sustainability Data` operation to retrieve energy management related metrics on a specific time range. Systems where the "environmental-metrics" feature is not available will only return a subset of the metrics (total-wattage, processor-utilization and ambient-temperature). This operation is supported using the BCPii interface. [Added by feature **environmental-metrics**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/get-historical-sustainability-data
```

In this request, the URI variable *{cpc-id}* is the Object ID of the CPC object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field Name	Type	Rqd/Opt	Description
`range`	String Enum	Optional	Time range for the historical data points. This is the amount of time to be covered by all data points. The possible values are as follows: • **"last-day"** - Last 24 hours. • **"last-week"** - Last 7 days. • **"last-month"** - Last 30 days. • **"last-three-months"** - Last 90 days. • **"last-six-months"** - Last 180 days. • **"last-year"** - Last 365 days. • **"custom"** - From **custom-range-start** to **custom-range-end**. If not specified, the default value is **"last-week"**.
`custom-range-start`	Timestamp	Required if **range** is **"custom"**	Start time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0.
`custom-range-end`	Timestamp	Required if **range** is **"custom"**	End time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than custom-range-start.
`resolution`	String Enum	Optional	Resolution of requested data points. This is the time interval in between data points. For systems where the "environmental-metrics" feature is not available, the minimum resolution is **"one-hour"**. The possible values are as follows: • **"fifteen-minutes"**- 15 minutes. • **"one-hour"**- 60 minutes. • **"one-day"**- 24 hours. • **"one-week"** - 7 days. • **"one-month"** - 30 days. If not specified, the default value is **"one-hour"**.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
`total-wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the total system power, in Watts, at a specific point in time.
`partition-wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the power consumption sum of all partitions, in Watts, at a specific point in time.
`infrastructure-wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the total infrastructure power (power consumed by Support Element, Ethernet switches, and cooling) in Watts, at a specific point in time.
`unassigned-wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the total power consumption of resources not assigned to a partition, in Watts, at a specific point in time.
`heat-load`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the amount of heat in Btu/h removed from the system per hour, at a specific point in time.
`heat-load-forced-air`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the amount of heat per hour removed from the system by forced-air, at a specific point in time.
`processor-utilization`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing processor utilization in percentage, at a specific point in time.
`ambient-temperature`	Array of float-data-point objects	Each element of this array is a float-data-point object representing temperature in degrees Celsius as measured by the system, at a specific point in time.
`exhaust-heat-temperature`	Array of float-data-point objects	Each element of this array is a float-data-point object representing exhaust air temperature in degrees Celsius, at a specific point in time.
`dew-point`	Array of float-data-point objects	Each element of this array is an integer-data-point object representing dew point in degrees Celsius, at a specific point in time.
`ambient-humidity`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the amount of water vapor in percentage, at a specific point in time.

Each nested integer-data-point object contains the following fields:

Table 517. integer-data-point nested object

Field Name	Type	Description
`data`	Integer	Stored property value at a specific time.
`timestamp`	Timestamp	This is specified as the number of milliseconds since the epoch.

Each nested float-data-point object contains the following fields:

Table 518. float-data-point nested object		
Field Name	**Type**	**Description**
`data`	Float	Stored property value at a specific time.
`timestamp`	Timestamp	This is specified as the number of milliseconds since the epoch.

Description

This operation returns an array of available historical power and environmental property data points in the CPC designated by *{cpc-id}*.

If the **range** field in the request body content is not **"custom"**, **custom-range-start** and **custom-range-end** are ignored and can be omitted from the request. Otherwise, those fields need to be set or HTTP status code 400 (Bad Request) is returned. Additionally, both need to be greater than zero or HTTP status code 400 (Bad Request) will be returned. Finally, **custom-range-end** must be greater than **custom-range-start** or else HTTP status code 400 (Bad Request) is returned. Should the custom range be greater than the existing range of measured data, the operation will complete successfully and return an array with the existing data points.

On successful execution, HTTP status code 200 (OK) is returned with the response body containing properties defined in "Response body contents" on page 1153. Systems where the "environmental-metrics" feature is not available on the CPC will only return a subset of those properties (total-wattage, processor-utilization and ambient-temperature), with a minimum resolution of one-hour. Should the "environmental-metrics" feature not be available on the HMC, HTTP status code 404.1 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- Action/task permission to the **Environmental Dashboard** task.

For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1153.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected or its value is not in the permitted range.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.
	4	The object designated by the request URI does not support the requested operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/b6fd4d2a-d18c-3af3-a467-175153b4bd84/operations/get-historical-sustainability-
data HTTP/1.1
x-api-session: pg1r7ahcfqjgan6t5c4nqugvnidxfv2bht24nltxo30oc78xk
Content-Type: application/json
Content-Length: 46
{
   "range":"last-day",
   "resolution":"one-day"
}
```

Figure 624. Get CPC Historical Sustainability Data: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 15 May 2023 16:23:10 GMT
Content-Type: application/json
Content-Length: 680
{
   "ambient-humidity":[
      {
         "data":50,
         "timestamp":1684109340806
      }
   ],
   "ambient-temperature":[
      {
         "data":15.0,
         "timestamp":1684109340806
      }
   ],
   "dew-point":[
      {
         "data":9.0,
         "timestamp":1684109340806
      }
   ],
   "exhaust-heat-temperature":[
      {
         "data":22.0,
         "timestamp":1684109336217
      }
   ],
   "heat-load":[
      {
         "data":32628,
         "timestamp":1684109333011
      }
   ],
   "heat-load-forced-air":[
      {
         "data":32628,
         "timestamp":1684109333013
      }
   ],
   "infrastructure-wattage":[
      {
         "data":804,
         "timestamp":1684109336255
      }
   ],
   "partition-wattage":[
      {
         "data":4178,
         "timestamp":1684109336255
      }
   ],
   "processor-utilization":[
      {
         "data":12,
         "timestamp":1684109333008
      }
   ],
   "total-wattage":[
      {
         "data":9550,
         "timestamp":1684109338353
      }
   ],
   "unassigned-wattage":[
      {
         "data":4571,
         "timestamp":1684109336255
      }
   ]
}
```

Figure 625. Get CPC Historical Sustainability Data: Response

CPC Install and Activate

The `CPC Install and Activate` operation installs and activates firmware on a specific CPC. [Added by feature **cpc-install-and-activate**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/install-and-activate
```

In this request, the URI variable *{cpc-id}* is the Object ID of the target CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`bundle-level`	String	Optional	Name of bundle to be installed. The **bundle-level** field cannot be present in the request body if the **ec-levels** field is present.
`ec-levels`	Array of ec-level objects	Optional	Array of nested **ec-level** objects (defined in the next table) that indicate the engineering change levels to be installed. The **ec-levels** field cannot be present in the request body if the **bundle-level** field is present.
`install-disruptive`	Boolean	Optional	Indicates if disruptive changes should be installed. If **true**, all firmware will be installed regardless of whether it is disruptive to CPC operations. If **false** and **bundle-level** or **ec-levels** is specified, the request will fail if the operation encounters a disruptive change. If **false**, and neither **bundle-level** or **ec-levels** are specified, all concurrent changes will be installed, and the disruptive ones will be left uninstalled. The **install-disruptive** field may not be present in the request body if the **bundle-level** field is present. Default value: **false**

An **ec-level** object contains information about a single Microcode Level (MCL) associated with an Engineering Change (EC) stream. Each **ec-level** object contains the following fields:

Table 519. ec-level nested object

Name	Type	Req/Opt	Description
`number`	String (1-6)	Required	Engineering Change (EC) number.

Table 519. ec-level nested object (continued)			
Name	**Type**	**Req/Opt**	**Description**
`mcl`	String (1-3)	Optional	A three-character decimal string that identifies the target Microcode Level (MCL) for the EC identified by **number.** If the value is less than three characters, it is padded on the front with zeros. The decimal value must not be specified as all zeros. Doing so will result in an exception since a value of "000" is considered the current base code level for an EC stream. If the **mcl** field is omitted, all MCLs currently available for the EC stream will be included.

Response body contents

Once the install and activate request is accepted, the response body contains a JSON object with the following fields:

Name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1160. The **job-results** field is null when this operation is successful. When it is not successful or partially successful, the **job-results** field contains an object with the following field:

Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful, or a description of firmware updates that are still pending when the operation was partially successful.

Description

The `CPC Install and Activate` operation installs and activates firmware updates on a CPC. The firmware is segmented into different subsystems identified by Engineering Change (EC) numbers. Sets of firmware updates within a single EC are packaged together and assigned a Microcode Level (MCL). MCL packages are installed sequentially, so an MCL implies not only the firmware updates that were packaged with that MCL, but all of the MCLs that preceded it in the EC stream. MCLs from multiple ECs are packaged together and assigned a Bundle Level. If the **bundle-level** field is present in the request body, all MCLs contained in the target bundle that have not already been installed on the target CPC are installed. If the **ec-levels** field is present in the request body, it identifies a set of EC MCLs that are to be installed on the target CPC. If neither the **bundle-level** or **ec-levels** fields are present, all firmware that has been retrieved, but not yet installed, is installed on the target CPC. Both the **bundle-level** and **ec-levels** cannot be present in the request body.

Note that it is not possible to remove firmware updates with this operation. Specifying a **bundle-level** or **ec-levels** field value that targets firmware updates that are already installed will result in an error rather than a removal of the firmware down to the specified level. Firmware can only be removed on a CPC by navigating to the **Remove and activate changes** operation in the **Change Internal Code** task.

The vast majority of firmware updates can be installed without disrupting CPC operations. Rarely, a firmware update will be released that does. To avoid inadvertently affecting CPC operations, the API client must explicitly give permission to install disruptive updates by specifying the **install-disruptive** field with a value of **true**. If a value of **false** is specified, or if the **install-disruptive** field is not present in the request body, and the targeted firmware contains a disruptive change, the behavior differs based on whether specific change levels have been specified. If the **bundle-level** or **ec-levels** field is present in the request body, the asynchronous job will fail immediately. If neither the **bundle-level** or **ec-levels** fields are present, all of the concurrent firmware updates will be installed and the disruptive ones will be left uninstalled. Disruptive updates cannot be installed by bundle, so the **install-disruptive** field my not be present in the request body if the **bundle-level** field is present.

If the API user does not have action/task permission to the **Change Internal Code** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{cpc-id}* does not identify a CPC object on the Console to which the API user has object-access permission. A 409 (Conflict) status code is returned if the target CPC is busy on the Console. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

If the request body fails to validate, a 400 (Bad Request) status code is returned. This could be because the **bundle-level** field is present in the request body and either the **ec-levels** or **install-disruptive** fields are also present.

If the request body contents are valid, a 202 (Accepted) response is returned and an asynchronous job is started to install the firmware updates identified by the **bundle-level** or **ec-levels** request body fields. Once started, the asynchronous job performs additional validation of the request body fields. If the **bundle-level** field is present in the request body but the identified bundle does not exist or is known but not yet retrieved or is already installed on the target CPC, or if the **ec-levels** field is present in the request body and references an Engineering Change (EC) number or Microcode Level (MCL) that does not exist or is already installed on the target CPC, a 400 (Bad Request) is returned in response to a Query Job Status Request. A 409 (Conflict) is returned if Change Management is not enabled, or if a connection to the Support System is not available on the target CPC, or if the **ec-levels** field identifies microcode levels that have dependencies on microcode levels that are not specified.

If the request body contents are valid, the firmware updates identified by the **bundle-level** or **ec-levels** request body fields are installed. If all updates are installed successfully, they are activated, which includes a restart of the target CPC's Support Element. If an error occurred when installing updates, any updates that were successfully installed are rolled back.

When the asynchronous job competes, the response to a Query Job Status request will include a **status** of **"complete"**. If the operation was successful, the completion status will be 204 (No Content). It is possible that additional actions, such as configuring virtual adapters off and on, may be required to fully activate certain firmware updates. If this is the case, the asynchronous job completion status will be 200 (OK), indicating overall success of the operation, but the result will contain a **job-results** field with a message indicating some firmware updates remain in a pending state. The View Internal Code Changes Summary task on the Console or Support Element user interface will provide a list of the additional actions that are required. It is not possible to query this information via the API. It is also possible to get a 200 (OK) status if the targeted firmware contains a change that is disruptive to CPC operations. In this case, all of the MCLs that could be concurrently installed were installed, and the disruptive MCLs were left uninstalled. The **job-results** field message will indicate which MCLs were left uninstalled.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is *{cpc-id}*.
- Action/task permission for the **Change Internal Code** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1158.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	15	The **bundle-level** and **ec-levels** fields are both present in the request body.
	15	The **bundle-level** and **install-disruptive** fields are both present in the request body
403 (Forbidden)	1	The API user does not have action/task permission to the **Change Internal Code** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	2	The CPC object with the **object-id** *{cpc-id}*was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the Console is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
200 (OK	N/A	The operation completed successfully, but some firmware remains in a pending state.
204 (No Content)	N/A	The operation completed successfully.

HTTP error status code	Reason code	Description
400 (Bad Request	354	The bundle identified by **bundle-level** does not exist on the target CPC.
	355	The bundle identified by **bundle-level** is known on the target CPC, but has not yet been retrieved.
	356	The bundle identified by **bundle-level** is already installed on the target CPC.
	357	There are installed MCLs currently installed on the target CPC that are beyond the bundle identified by **bundle-level.**
	358	The bundle identified by **bundle-level** has been marked as invalid for install.
	359	The bundle identified by **bundle-level** contains one or more updates that are disruptive to CPC operations and the **install-disruptive** field was omitted or specified with a value of **false**.
	366	There are MCL alerts present for install and activate of the bundle identified by bundle-level. Log on to the GUI of the Console managing the target CPC, then navigate to the **Change Internal Code** task and rerun the operation.
	378	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that does not identify a known component and is therefore invalid.
	380	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that is disruptive to CPC operations. This cannot occur since the **install-disruptive** field was omitted or specified with a value of **false.**
409 (Conflict)	341	The CPC identified by *{cpc-id}* does not have an active connection to the Support System.
	367	Change Management is not enabled on the CPC identified by *{cpc-id}*.
	383	There are no internal code changes on the system, so the change internal code operation could not be performed.
	385	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that is not properly bounded by the current applied and staged levels.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/d1f5e333-f995-3ce8-94e8-2e2fdffd2c94/operations/install-and-activate HTTP/1.1
x-api-session: 329lt0xwj1pfakc423x2wfjs26zygbmh1qq6l22rj2fyz39c6f
Content-Type: application/json
Content-Length: 51
{
   "ec-levels":[
      {
         "mcl":"001",
         "number":"P30719"
      }
   ]
}
```

Figure 626. CPC Install and Activate: Request

```
202
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/05b7b378-b455-11ed-8eb6-fa163ecad2ab"
}
```

Figure 627. CPC Install and Activate: Response

CPC Delete Retrieved Internal Code

The `CPC Delete Retrieved Internal Code` operation deletes retrieved internal code that has not been installed on a specific CPC. [Added by feature **cpc-delete-retrieved-internal-code**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/delete-retrieved-internal-code
```

In this request, the URI variable *{cpc-id}* is the Object ID of the target CPC.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`ec-levels`	Array of ec-level objects	Optional	Array of nested **ec-level** objects (defined in Table 519 on page 1157) that indicate the uninstalled engineering change levels to be deleted. Default: All retrieved, uninstalled MCLs are deleted down to the applied levels.

Response body contents

Once the `CPC Delete Retrieved Internal Code` request is accepted, the response body contains a JSON object with the following fields

Name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent, and results are available for the asynchronous portion of this operation. These results are retrieved using the Query Job Status operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the **status** of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Response body contents" on page 1162. The **job-results** field is null when this operation is successful. When it is not successful or partially successful, the **job-results** field contains an object with the following field:

Name	Type	Description
`message`	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

The `CPC Delete Retrieved Internal Code` operation deletes retrieved internal code that has not been installed on a CPC. The firmware is segmented into different subsystems identified by Engineering Change (EC) numbers. Sets of firmware updates within a single EC are packaged together and assigned a Microcode Level (MCL). MCL packages are installed sequentially, so an MCL implies not only the firmware updates that were packaged with that MCL, but all of the MCLs that preceded it in the EC stream. If the **ec-levels** field is present in the request body, it identifies a set of retrieved, but uninstalled EC MCLs that are to be deleted on the target CPC. If the **ec-levels** field is not present, then all the firmware that is currently retrieved, but not installed, will be deleted from the target CPC.

If the API user does not have action/task permission to the **Change Internal Code** task, a 403 (Forbidden) status code is returned. A 404 (Not Found) status code is returned if *{cpc-id}* does not identify a CPC object on the Console to which the API user has object-access permission. A 409 (Conflict) status code is returned if the target CPC is busy on the Console. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

If the request body fails to validate, a 400 (Bad Request) status code is returned.

If the request body contents are valid, a 202 (Accepted) response is returned and an asynchronous job is started to remove the firmware updates identified by the **ec-levels** request body field. Once started, the asynchronous job performs additional validation of the request body fields. If the **ec-levels** field is present in the request body and references an Engineering Change (EC) number or Microcode Level (MCL) that does not exist on the target CPC, a 400 (Bad Request) is returned in response to a Query Job Status Request. A 409 (Conflict) is returned if Change Management is not enabled, or if a connection to the Support System is not available on the target CPC, or if the **ec-levels** field identifies microcode levels that are already installed on the system instead of only being retrieved.

If the request body contents are valid, the firmware updates identified by the **ec-levels** request body field are deleted. If an error occurred when deleting the updates, then only the updates that were unsuccessfully deleted will remain on the system; any updates that were deleted before reaching an error will remain deleted upon completion of the operation.

When the asynchronous job competes, the response to a Query Job Status request will include a status of "complete". If the operation was successful, the completion status will be 204 (No Content).

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC whose **object-id** is *{cpc-id}*.
- Action/task permission for the **Change Internal Code** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Job status and reason codes" on page 1164.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission to the **Change Internal Code** task.
404 (Not Found)	1	A CPC with the **object-id** *{cpc-id}* does not exist on the Console or the API user does not have object-access permission for it.
409 (Conflict)	2	The CPC object with the **object-id** *{cpc-id}*was busy and the request timed out.
503 (Service Unavailable)	1	The request could not be processed because the Console is not currently communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	The operation completed successfully.
400 (Bad Request	378	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that does not identify a known component and is therefore invalid.
409 (Conflict)	341	The CPC identified by *{cpc-id}* does not have an active connection to the Support System.
	367	Change Management is not enabled on the CPC identified by *{cpc-id}*.
	383	There are no internal code changes on the system, so the change internal code operation could not be performed.
	385	The **ec-levels** field contains an ec-level object with a **number** and **mcl** combination that is not properly bounded by the current applied and staged levels.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/d1f5e333-f995-3ce8-94e8-2e2fdffd2c94/operations/delete-retrieved-internal-code
HTTP/1.1
x-api-session: 5uxaze732c5oga0dm3iq0datfeinr6lzzrcy7qctbplp6b0i85
Content-Type: application/json
Content-Length: 51
{
   "ec-levels":[
      {
         "mcl":"001",
         "number":"P30719"
      }
   ]
}
```

Figure 628. CPC Delete Retrieved Internal Code: Request

```
202
Content-Type: application/json
Content-Length: 60
{
   "job-uri":"/api/jobs/a87178d4-b454-11ed-9903-fa163ecad2ab"
}
```

Figure 629. CPC Delete Retrieved Internal Code: Response

List CPC API Features

The `List CPC API Features` operation returns information about the API features available on the CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/list-features
```

Response body contents

On successful completion, the response body is a JSON array of String values, each of which identifies an available API feature. The order in which these strings are returned is unspecified. The possible feature names are listed in “API features” on page 103.

Description

This operation lists the API features available on the CPC. Beginning with API version 4.10, API clients must use this operation and the `List Console API Features` operation to determine if specific new or changed API functionality is available.

If the **name** query parameter is specified, the returned list is limited to those API features that have a **name** field matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed

Authorization requirements

This operation has no explicit authorization requirements; however, the request must contain the session ID of a fully-authenticated API session.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned, and the response body is provided as described in "Response body contents" on page 1165.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/1d01b986-d466-3ff8-8937-685cd04169a2/operations/list-features HTTP/1.1
x-api-session: 5ea3ict7u3h5iwwvmnxojj4n1faz7y4w71aluhidnxqrfky52l
```

Figure 630. List CPC API Features: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 31 Jan 2023 13:59:06 GMT
Content-Type: application/json
Content-Length: 140
[
   "environmental-metrics",
   "cpc-install-and-activate",
   "secure-boot-with-certificates",
   "report-a-problem",
   "dpm-smcd-partition-link-management"
]
```

Figure 631. List CPC API Features: Response

Usage notes

- Although object-access permission to the target CPC is not required by this operation, the request URI must contain a valid *{cpc-id}*. An API user with object-access permission to at least one Partition or Logical Partition object can obtain the URI of its parent CPC and use it to construct the request URI for this operation. The parent CPC URI is provided by the following operations: `Get Partition Properties`, `Get Permitted Partitions`, `Get Logical Partition Properties`, `Get Permitted Logical Partitions`, and `Get Inventory`.
- Since the API feature names returned by this operation pertain to the SE associated with the CPC, the feature information is only useful to API users with permission to issue operations relevant to the CPC or its partitions or other related objects.

Switch Support Elements

The `Switch Support Elements` operation allows the roles of the Primary Support Element and the Alternate Support Element to be switched. [Added by feature **switch-support-elements**].

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/switch-support-elements
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve job status.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the Query Job Status operation is specified in the description for the Query Job Status operation. When the status of the job is "**complete**", the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1169. The **job-results** field is null when this operation is successful. When it is not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was not successful.

Description

Switch Support Elements switches the roles of the two Support Elements:

- The Primary Support Element reboots and becomes the Alternate Support Element
- The Alternate Support Element reboots and becomes the Primary Support Element

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent, with Job Status and Reason Codes described in "Job status and reason codes" on page 1169.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the CPC object designated by *{cpc-id}*
- Action/task permission for the **Alternate Support Element** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1167.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	386	Primary Support Element state information was not successfully received at Alternate Support Element.
	387	Support Element is fenced from previous automatic switchover
	388	Engineering Changes is in progress.
	389	Hard disk Restore is in progress.
	390	Change Internal Code is applying Licensed Internal Code changes.
	391	Mirroring to the Alternate Support Element is in progress.
	392	Alternate Support Element Licensed Internal Code is different than that of the Primary Support Element.
	393	Alternate Support Element has system status check.
	394	Primary Support Element cannot communicate with the Alternate Support Element
	395	Alternate Support Element is preloaded for disruptive switch activation of a new Engineering Changes level.
	396	Alternate Support Element is preloaded for concurrent activation of a new Engineering Changes level.
	397	ISO Mirror is in progress.
	398	Alternate Support Element information was not successfully retrieved.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status code	Job reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/2f6ed05f-adde-3e06-a180-02f8b80e23bb/operations/switch-support-elements HTTP/1.1
x-api-session: 48dghpt7k4ozci8qtb5zrhxxisuzvyasp9oj62esh2gjujnvlx
Content-Type: application/json
```

Figure 632. Switch Support Elements: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Wed, 29 Nov 2023 00:33:27 GMT
<No response body>
```

Figure 633. Switch Support Elements: Response

Inventory service data

Information about CPCs can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the CPC objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular CPC is included only if the API user has access permission to that object as described in the `Get CPC Properties` operation.

For each CPC object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for “Get CPC Properties” on page 1039. That is, the data provided is the same as would be provided if a `Get CPC Properties` operation were requested targeting this object.

Logical Partition object

The Processor Resource/Systems Manager (PR/SM) is a feature of IBM mainframes that enables logical partitioning of the CPC. A logical partition (LPAR) is a virtual machine at the hardware level. Each LPAR operates as an independent server running its own operating environment. Each LPAR runs its own operating system, which can be any mainframe operating system.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see “Property characteristics” on page 98.

This object includes the properties defined in "Base managed object properties schema" on page 100, with the following class-specific specialization:

Table 520. Logical Partition object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Logical Partition object, of the form `/api/logical-partitions/{logical-partition-id}` where *{logical-partition-id}* is the value of the **object-id** property of the Logical Partition object.
parent	—	String/ URI	The canonical URI path of the associated CPC object.
class	—	String	The class of a Logical Partition object is **"logical-partition"**.
name	(ro)(pc)	String (1-8)	The name of the logical partition
description	(ro)	String (0-1024)	The descriptive text associated with this object.
status	(sc)	String Enum	The current operational status of the LPAR object. One of the following values: • **"operating"** - all processors on the LPAR object are operating. • **"not-operating"** - all processors on the LPAR object are stopped. • **"not-activated"** - the LPAR is not activated. • **"exceptions"** - one or more processors on the LPAR object are operating and one or more are not operating.
acceptable-status	(w)(pc)	Array of String Enum	The set of operational status values in which the LPAR object can exist and be considered in an acceptable (not alert causing) state. For HMC (available via WSAPI), zero or more of: • **"operating"** - all processors on the LPAR object are operating. • **"not-operating"** - all processors on the LPAR object are stopped. • **"not-activated"** - the LPAR is not activated. • **"exceptions"** - one or more processors on the LPAR object are operating and one or more are not operating. For SE (available via BCPii), zero or more of: • **"operating"** - all processors on the LPAR object are operating. • **"not-operating"** - all processors on the LPAR object are stopped. • **"not-activated"** - the LPAR is not activated. • **"exceptions"** - one or more processors on the LPAR object are operating and one or more are not operating. • **"acceptable"** - indicates all channels in the LPAR object are not operating, but their statuses are acceptable.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties. Refer to the *Processor Resource/Systems Manager Planning Guide* for more detailed explanations of the various properties.

There are additional notes throughout the table. Please refer to the note list at the end of the table.

Table 521. Logical Partition object: class specific additional properties

Name	Qualifier	Type	Description
cpc-name	(a)	String	The **name** property of the logical partition's parent CPC object.
se-version	(a)	String	The **se-version** property of the logical partition's parent CPC object.
os-name[1]	(pc)	String (0-8)	An operating system provided value, used to identify the operating system instance. The format of the value is operating system dependent. If not provided by the operating system, an empty string is returned.
os-type[1]	(pc)	String (0-8)	A human readable form of the operating system provided value for the type of the operating system active in this logical partition. If not provided, an empty string is returned.
os-level[1]	(pc)	String (0-32)	A human readable form of the operating system provided value for the level of the operating system active in this logical partition. If not provided, an empty string is returned.
sysplex-name[1]	(pc)	String (1-8)	Applicable only for z/OS or when **os-type** is **"CFCC"**, the name of the sysplex of which this logical partition is a member, or an empty string if the logical partition is not a member of a sysplex. For a logical partition that is not z/OS and **os-type** is not **"CFCC"**, a null object is returned. When there is no operating system loaded in this logical partition, a null object is returned.
is-sub-capacity-boost-active[1, 10]	(pc)	Boolean	If true, sub-capacity boost is active. If false, sub-capacity boost is inactive.
is-secure-execution-enabled[1, 10]	(pc)	Boolean	If true, Secure Execution for Linux is enabled. If false, Secure Execution for Linux is not enabled.
is-ziip-capacity-boost-active[1, 10]	(pc)	Boolean	If true, zIIP-capacity boost is active. If false, zIIP-capacity boost is inactive.
speed-boost[14]	(pc)	boost-info object	The nested boost-info object containing information about additional processing capacity via general purpose processors as described in Table 522 on page 1193. If there is no general purpose processor allocated or reserved for the logical partition, null is returned.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ziip-boost[14]	(pc)	boost-info object	The nested boost-info object containing information about additional processing capacity via zIIP processors as described in Table 522 on page 1193. If there is no zIIP processor allocated or reserved for the logical partition, null is returned.
has-operating-system-messages[1]	—	Boolean	If true, object has operating system messages. If false, object does not have operating system messages.
has-important-unviewed-operating-system-messages	—	Boolean	If true, object has unviewed operating system messages requiring attention. If false, object does not have unviewed operating system messages requiring attention.
activation-mode	(pc)	String Enum	One of the following values: • **"general"** - the logical partition is in general mode • **"esa390"** - the logical partition is in ESA/390 mode • **"esa390tpf"** - the logical partition is in ESA/390 TPF mode • **"coupling-facility"** - the logical partition is running as a coupling facility • **"linux"** - the logical partition is in Linux mode • **"zvm"** - the logical partition is in z/VM mode • **"zaware"** - the logical partition is in IBM zAware mode • **"ssc"** - the logical partition is in IBM Secure Service Container mode • **"not-set"** - the logical partition is not activated.
next-activation-profile-name	(w)(pc)	String (1-16)	Image activation profile name or load activation profile name to be used on the next activate. The **group-uri** query parameter can be used on a `Get Logical Partition Properties` operation to specify the object URI of the Custom Group object used for determining the next activation profile name to be used. If not specified, the system-defined Logical Partition group is used for this determination.
last-used-activation-profile	(pc)	String (0-16)	The last used activation profile name or an empty string.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
last-used-load-type	(pc)	String Enum	The type of load operation most recently done in the logical partition or **"standard"** if the value is not available.[15] One of: • **"ipltype-standard"** – standard load • **"ipltype-scsi"** – SCSI load • **"ipltype-scsidump"** – SCSI dump • **"ipltype-nvmeload"** – NVMe load • **"ipltype-nvmedump"** – NVMe dump • **"ipltype-tape-load"** - A Channel Command Word (CCW) tape OS load [Added by feature **secure-boot-with-certificates**] • **"ipltype-tape-dump"** - A Channel Command Word (CCW) tape dump [Added by feature **secure-boot-with-certificates**] • **"ipltype-eckd-ccw-load"** - A Channel Command Word (CCW) ECKD OS load • **"ipltype-eckd-ccw-dump"** - A Channel Command Word (CCW) ECKD dump [Added by feature **secure-boot-with-certificates**] • **"ipltype-eckd-ld-load"** - A list-directed ECKD OS load [Added by feature **secure-boot-with-certificates**] • **"ipltype-eckd-ld-dump"** - A list-directed ECKD dump [Added by feature **secure-boot-with-certificates**] • **"ipltype-standard"** - A standard load. This value is only returned when the SE version is 2.15.0 or earlier, is 2.16.0 and the **secure-boot-with-certificates** feature is not available on the CPC, or if a load has not yet been completed. [Updated by feature **secure-boot-with-certificates**] Default: **"ipltype-standard"**

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
last-used-load-address	(pc)	String (4-5)	The load address most recently used by the logical partition or "00000" if the value is not available.[15] When **last-used-load-type** is **"ipltype-standard"**, **"ipltype-scsi"**, or **"ipltype-scsidump"**, the load address is a hexadecimal string of the form *cdddd* where *c* is the channel subsystem ID and *dddd* is the device address of the I/O device that was loaded. When **last-used-load-type** is **"ipltype-nvmeload"** or **"ipltype-nvmedump"**, the load address is a 4-digit hexadecimal string (i.e. the function ID, or FID) of an I/O device that provided access to the control program that was loaded. Otherwise, the load address is a hexadecimal string of the form *cdddd* where *c* is the channel subsystem IB and *dddd* is the device address of the I/O device that was loaded. Default: "00000" [Updated by feature **secure-boot-with-certificates**]
last-used-load-parameter	(pc)	String (0-8)	The load parameter most recently used by the logical partition or null if the value is not available. Default: An empty string
last-used-secure-boot[10]	(pc)	Boolean	Verify software signature with distributor most recently used by the logical partition or null if the value is not available. Default: false
last-used-world-wide-port-name	(pc)	String (1-16)	The worldwide port name (WWPN), in hexadecimal, of the target SCSI device most recently used by the logical partition or null if the value is not available. Default: "0"
last-used-logical-unit-number	(pc)	String (1-16)	The logical unit number (LUN), in hexadecimal, of the target SCSI device most recently used by the logical partition or null if the value is not available. Default: "0"
last-used-disk-partition-id	(pc)	Integer (0-30)	The **disk-partition-id** (also called the boot program selector) of the target SCSI, ECKD, or NVMe device most recently used by the logical partition or 0 if the value is not available.[15] This value is indeterminate if the **last-used-disk-partition-id-automatic** value is **true**. Default: 0 [Updated by feature **secure-boot-with-certificates**]

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
last-used-operating-system-specific-load-parameters	(pc)	String (0-256)	The operating system specific load parameters of the target SCSI, ECKD, or NVMe device most recently used by the logical partition or an empty string if the value is not available.[15] Default: An empty string [Updated by feature **secure-boot-with-certificates**]
last-used-boot-record-location-cylinder[16]	(pc)	String (1-7)	The most recently used boot record location cylinder value in hexadecimal, or null if the value is not available.[15] This value is indeterminate if the **last-used-boot-record-location-volume-label** is **true**. [Added by feature **secure-boot-with-certificates**]
last-used-boot-record-location-head[16]	(pc)	String (1)	The most recently used boot record location head value in hexadecimal, or null if the value is not available.[15] This value is indeterminate if the **last-used-boot-record-location-volume-label** is **true**. [Added by feature **secure-boot-with-certificates**]
last-used-boot-record-location-record[16]	(pc)	String (1-2)	The most recently used boot record location record value in hexadecimal, or null if the value is not available.[15] This value is indeterminate if the **last-used-boot-record-location-volume-label** is **true**. [Added by feature **secure-boot-with-certificates**]
last-used-boot-record-location-volume-label[16]	(pc)	Boolean	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** were determined by the volume label, or null if the value is not available.[15] If this value is **true**, the values in **last-used-boot-record-location-cylinder**, **last-used-boot-record-location-head**, and **last-used-boot-record-location-record** are indeterminate. [Added by feature **secure-boot-with-certificates**]
last-used-device-type[16]	(pc)	String Enum	The load device type most recently used by the logical partition or null if the value is not available.[15] One of: • **"eckd"** - An ECKD load device • **"nvme"** - An NVMe load device • **"scsi"** - A SCSI load device • **"tape"** - A tape load device [Added by feature **secure-boot-with-certificates**]

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
last-used-load-program-type[16]	(pc)	String Enum	The program type last loaded by the logical partition, or null if the value is not available.[15] One of: • **"os"** - An operating system • **"dump"** - A dump program • [Added by feature **secure-boot-with-certificates**]
last-used-operation-type[16]	(pc)	String Enum	The type of load operation last used by the logical partition for an ECKD load, or null if the value is not available.[15] One of: • **"list-directed"** - A list-directed operation • **"ccw"** - A Channel Command Word operation [Added by feature **secure-boot-with-certificates**]
last-used-disk-partition-id-automatic[16]	(pc)	Boolean	The method used to determine the **disk-partition-id** (also known as the boot program selector) for the most recent load operation. **True** if the last load used an automatic **disk-partition-id**, **false** if it was manually determined, or null if the value is unavailable.[15] If this field is **true**, then the value in **last-used-disk-partition-id** is indeterminate. [Added by feature **secure-boot-with-certificates**]
last-used-boot-record-logical-block-address	(pc)	String (1-16)	The boot record logical block address, in hexadecimal, of the target SCSI or NVMe device most recently used by the logical partition or null if the value is not available. Default: "0"
last-used-clear-indicator	(pc)	Boolean	The clear indicator most recently used by the logical partition, or null if the value is not available. Default: true
initial-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer (1-999)	The relative amount of shared general purpose processor resources allocated to the logical partition.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
initial-processing-weight-capped[1, 2, 3, 4]	(w)(pc)	Boolean	Whether the initial processing weight for general purpose processors is a limit or a target. **True** Indicates that the initial general purpose processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources. **False** Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.
minimum-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be less than or equal to the initial-processing-weight property.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
maximum-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Defines the maximum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be greater than or equal to the initial-processing-weight property.
current-processing-weight[1, 3]	(pc)	Integer (1-999)	The relative amount of shared general purpose processor resources currently allocated to the logical partition.
current-processing-weight-capped[1, 2, 3]	—	Boolean	Whether the current general purpose processing weight is a limit or a target. **True** Indicates that the current general purpose processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current general purpose processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
workload-manager-enabled[1, 5, 9]	(w)(pc)	Boolean	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. **True** Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. **False** Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition. **Note:** When the logical partition is in an activation mode that does not support Workload Manager (e.g. Coupling Facility), including this property in the Update request may not have any effect.
absolute-processing-capping	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the general purpose processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
defined-capacity[1]	(w)	Integer	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed to by z/OS Workload Manager for the purpose of software pricing. **0** No defined capacity is specified for this logical partition. **1-nnnn** Represents the amount of defined capacity specified for this logical partition.
cluster-name[1]	(pc)	String (0-8)	LPAR cluster name, which identifies membership in a group of logical partitions that are members of the same z/OS Parallel Sysplex®.
partition-number[1]	(pc)	String (2)	The partition number for the logical partition, in hexadecimal.
partition-identifier[1]	(pc)	String (2)	The partition identifier for the logical partition, in hexadecimal.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
processor-usage[1, 10, 11]	(pc)	String Enum	How processors are allocated to the logical partition. One of the following values: • **"dedicated"** - all processor types in the logical partition are to be exclusively available to this specific logical partition. • **"shared"** - all processor types in the logical partition are to be shareable across logical partitions.
number-general-purpose-processors[1, 10, 11]	—	Integer	The number of general purpose processors allocated to the logical partition.
number-reserved-general-purpose-processors[1, 10, 11]	—	Integer	The number of general purpose processors reserved for the logical partition.
number-general-purpose-cores[1, 10, 11, 12, 13]	(w)(pc)	Integer	The number of general purpose processor cores allocated to the logical partition.
number-reserved-general-purpose-cores[1, 10, 11, 12, 13]	(w)	Integer	The number of general purpose processor cores reserved for the logical partition.
number-icf-processors[1, 10, 11]	—	Integer	The number of Internal Coupling Facility (ICF) processors allocated to the logical partition.
number-reserved-icf-processors[1, 10, 11]	—	Integer	The number of Internal Coupling Facility (ICF) processors reserved for the logical partition.
number-icf-cores[1, 10, 11, 12, 13]	(w)(pc)	Integer	The number of Internal Coupling Facility (ICF) processor cores allocated to the logical partition.
number-reserved-icf-cores[1, 10, 11, 12, 13]	(w)	Integer	The number of Internal Coupling Facility (ICF) processor cores reserved for the logical partition.
number-ifl-processors[1, 10, 11]	—	Integer	The number of Integrated Facility for Linux (IFL) processors allocated to the logical partition.
number-reserved-ifl-processors[1, 10, 11]	—	Integer	The number of Integrated Facility for Linux (IFL) processors reserved for the logical partition.
number-ifl-cores[1, 10, 11, 12, 13]	(w)(pc)	Integer	The number of Integrated Facility for Linux (IFL) processor cores allocated to the logical partition.
number reserved-ifl-cores[1, 10, 11, 12, 13]	(w)	Integer	The number of Integrated Facility for Linux (IFL) processor cores reserved for the logical partition.
number-ziip-processors[1, 10, 11]	—	Integer	The number of z Integrated Information Processor (zIIP) processors allocated to the logical partition.
number-reserved-ziip-processors[1, 10, 11]	—	Integer	The number of z Integrated Information Processor (zIIP) processors reserved for the logical partition.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
number-ziip-cores[1, 10, 11, 12, 13]	(w)(pc)	Integer	The number of z Integrated Information Processor (zIIP) processor cores allocated to the logical partition.
number-reserved-ziip-cores[1, 10, 11, 12, 13]	(w)	Integer	The number of z Integrated Information Processor (zIIP) processor cores reserved for the logical partition.
initial-aap-processing-weight[1, 2, 3, 9]	(w)	Integer (1-999)	The relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
initial-aap-processing-weight-capped[1, 2, 3, 4]	(w)	Boolean	Whether the initial processing weight for Application Assist Processor (zAAP) processors is a limit or a target. **True** Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Application Assist Processor (zAAP) processor resources, regardless of the availability of excess Application Assist Processor (zAAP) processor resources. **False** Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is not capped. It represents the share of Application Assist Processor (zAAP) processor resources guaranteed to a logical partition when all Application Assist Processor (zAAP) processor resources are in use. Otherwise, when excess Application Assist Processor (zAAP) processor resources are available, the logical partition can use them if necessary.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
minimum-aap-processing-weight[1, 2, 3, 9]	(w)	Integer	The minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Update: **0** No minimum value for the processing weight. **1-999** Define the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
maximum-aap-processing-weight[1, 2, 3, 9]	(w)	Integer	The maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
current-aap-processing-weight[1, 3]	—	Integer (1-999)	The current relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
current-aap-processing-weight-capped[1, 3]	—	Boolean	Whether the current Application Assist Processor (zAAP) processing weight is a limit or a target. **True** Indicates that the current Application Assist Processor (zAAP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current Application Assist Processor (zAAP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-aap-capping	(w)	absolute-capping object	The amount of absolute capping applied to the Application Assist Processor (zAAP). **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
initial-ifl-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer (1-999)	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
initial-ifl-processing-weight-capped[1, 2, 3, 4]	(w)(pc)	Boolean	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. **True** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources **False** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
minimum-ifl-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
maximum-ifl-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
current-ifl-processing-weight[1, 3]	(pc)	Integer (1-999)	The current relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
current-ifl-processing-weight-capped[1, 3]	—	Boolean	Whether the current Integrated Facility for Linux (IFL) processing weight is a limit or a target. **True** Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current Integrated Facility for Linux (IFL) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-ifl-capping	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
initial-ziip-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer (1-999)	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
initial-ziip-processing-weight-capped[1, 2, 3, 4]	(w)(pc)	Boolean	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. **True** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. **False** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
minimum-ziip-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
maximum-ziip-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
current-ziip-processing-weight[1, 3]	(pc)	Integer (1-999)	The current relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
current-ziip-processing-weight-capped[1, 3]	—	Boolean	Whether the current z Integrated Information Processors (zIIP) processing weight is a limit or a target. **True** Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current z Integrated Information Processors (zIIP) processing weight for the logical partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-ziip-capping	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
initial-cf-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer (1-999)	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
initial-cf-processing-weight-capped[1, 2, 3, 4]	(w)(pc)	Boolean	Indicates whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. **True** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. **False** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Logical Partition is not capped. It represents the share of Internal Coupling Facility (ICF) processor resources guaranteed to a logical partition when all Internal Coupling Facility (ICF) processor resources are in use. Otherwise, when excess Internal Coupling Facility (ICF) processor resources are available, the logical partition can use them if necessary.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
minimum-cf-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: **1-999** The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
maximum-cf-processing-weight[1, 2, 3, 9]	(w)(pc)	Integer	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.
current-cf-processing-weight[1, 3]	(pc)	Integer (1-999)	The current relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the Logical Partition object.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
current-cf-processing-weight-capped[1, 3]	—	Boolean	Indicates whether the current Internal Coupling Facility (ICF) processing weight is a limit or a target. **True** Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition object is capped. It represents the logical partition's maximum share of resources, regardless of the availability of excess processor resources. **False** Indicates that the current Internal Coupling Facility (ICF) processing weight for the Logical Partition is not capped. It represents the share of resources guaranteed to a logical partition when all processor resources are in use. Otherwise, when excess processor resources are available, the logical partition can use them if necessary.
absolute-cf-capping	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor. **Note:** Absolute capping does not apply to image profiles where the processors are dedicated to the partition. Absolute capping only applies to partitions using shared processors.
program-status-word-information[1]	—	Array of psw-description objects	Describes the current PSW information for each CP associated with the logical partition. The information is obtained on each Get Logical Partition Properties request and is not cached. Refer to the description of the psw-description object for details.
initial-vfm-storage[8]	—	Long	The initial amount of Virtual Flash Memory (VFM) storage, in gigabytes (GB), to be allocated to this logical partition at activation. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC.
maximum-vfm-storage[8]	—	Long	The maximum amount of VFM storage, in gigabytes (GB), that can be allocated to this logical partition while it is running. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC.
current-vfm-storage[8]	—	Long	The current amount of VFM storage, in gigabytes (GB), that is allocated to this logical partition. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC.

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
os-ipl-token[1]	(pc)	String (1-16)	Applicable only to z/OS, a value provided when z/OS is IPLed that uniquely identifies the instance of the operating system. Used by z/OS to obtain knowledge about the status of another system in the sysplex, and upon the demise of the system, potentially partition the system out of the sysplex immediately and reset the demised system. The value is a string of hexadecimal characters (0-9,A-Z), left justified.
group-profile-capacity[1]	—	Integer	The current value of the **effective-capacity** property of the Group Profile with which the logical partition is associated. A null object is returned if the logical partition is not assigned to an LPAR group.
group-profile-uri[1]	—	String/ URI	The canonical URI of the Group Profile associated with the logical partition. A null object is returned if the logical partition is not assigned to an LPAR group.
zaware-host-name[6]	(w)	String (1-64)	The IBM zAware host name. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and colon(:)
zaware-master-userid[6]	(w)	String (1-32)	The IBM zAware master userid. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and underscore (_)
zaware-master-pw[6]	(wo)	String (8-256)	The IBM zAware master password. Valid characters are: a-z,A-Z,0-9 and !@#$%^&*()_+{}|<>?-= This property is not returned on a Get request, it can only be specified on an Update request.
zaware-network-info[6]	(w)	Array of zaware-network objects	The set of networks available to IBM zAware. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
zaware-gateway-info[6]	(w)	ip-info object	The default gateway IP address information. A null object indicates no default gateway IP address is specified.
zaware-dns-info[6]	(w)	Array of ip-info objects	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.
ssc-host-name[7]	(w) or — if **se-version** is **"2.14.0"** or later	String (1-64)	The Secure Service Container name. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and colon(:).
ssc-master-userid[7]	(w) or — if **se-version** is **"2.14.0"** or later	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-), and underscore(_).

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ssc-master-pw[7]	(wo) or — if **se-version** is **"2.14.0"** or later	String (8-256)	The Secure Service Container master user password. Valid characters are: a-z, A-Z, 0-9, and !@#$%^&*()_+{}|<>?-=. This property is not returned on a Get request; it can be specified on an Update request.
ssc-network-info[7]	(w) or — if **se-version** is **"2.14.0"** or later	Array of ssc-network objects	The set of networks available to the Secure Service Container. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
ssc-gateway-info[7]	(w) or — if **se-version** is **"2.14.0"** or later	ip-info object	The default gateway IP address information for the Secure Service Container. A null object indicates no default gateway IP address is specified. Only IPv4 address types are supported.
ssc-dns-info[7]	(w) or — if **se-version** is **"2.14.0"** or later	Array of ip-info objects	The DNS IP address information for the Secure Service Container. A minimum of 0 entries and a maximum of 2 entries are permitted On an Update request, this property fully replaces the existing set.
storage-central-allocation	(pc)	Array of central-storage-allocation objects	A nested object that details the central storage allocated to the logical partition. Refer to Table 523 on page 1193 for details. If there are no central-storage-allocation objects, a null object is returned.
storage-expanded-allocation	(pc)	Array of expanded-storage-allocation objects	A nested object that details the expanded storage allocated to the logical partition. Refer to Table 524 on page 1194 for details. If there are no expanded-storage-allocation objects, a null object is returned.
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
request-origin	—	Boolean	If true, the logical partition object is the same as the origin of the request. If false, the logical partition is not the same as the origin of the request. **Note:** This property is only returned when the BCPii interface was used for the request.
assigned-certificate-uris	(c)(pc)	Array of String/ URI	Array of URIs referring to the certificates that are assigned to this logical partition, or an empty array if there are no assigned certificates. [Added by feature **secure-boot-with-certificates**]

Table 521. Logical Partition object: class specific additional properties (continued)			
Name	**Qualifier**	**Type**	**Description**

Notes:

1. If the logical partition **status** property is **"not-activated"**, a null object is returned instead of the documented field type.
2. An Update of this property is only valid for an **object-id** that represents a logical partition with at least one shared processor of the corresponding type.
3. The value returned from a Get request is a null object for an **object-id** that does not represent a logical partition with at least one shared processor of the corresponding type.
4. This property and the **workload-manager-enabled** property are mutually exclusive and cannot both be enabled at the same time. Therefore in order to enable this property it might be necessary to first disable the **workload-manager-enabled** property.
5. This property and the various capping properties are mutually exclusive and cannot be enabled at the same time. Therefore in order to enable this property it may be necessary to first disable any capping property that is currently enabled.
6. On a Get request, this property is returned only when **activation-mode** is **"zaware"**. On an Update request, this property can be updated only when **activation-mode** is **"zaware"**.
7. On a Get request, this property is returned only when **activation-mode** is **"ssc"**. On an Update request, this property can be updated only when **activation-mode** is **"ssc"**.
8. On a Get request, this property is returned only when the SE version is 2.14.0 or later.
9. On an Update request, when **workload-manager-enabled** is **true**, the initial processing weight must be greater than or equal to the minimum processing weight, and less than or equal to the maximum processing weight within the same processor type.
10. This property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
11. If the corresponding data is not available, a null object is returned.
12. On an Update request, the input value must be greater than the value that is currently in the logical partition.
13. On an Update request, if a new processor type is being defined (both the current numbers of the allocated and reserved cores of a processor type are 0, and at least one of them is being increased), the following conditions must be met:
    - The target logical partition is activated.
    - The activation mode of the target logical partition supports multiple processor types and the newly defined type is one of them.
    - If the current **processor-usage** is **"shared"**, the initial processing weight of the corresponding processor type is also included in the request.
14. This property is returned only when the associated SE version is 2.16.0 or later.
15. A value may not be available if a load has not previously been completed on this Logical Partition, or if the value was not specified for a previous load because it was not applicable to the **last-used-load-type**. [Added by feature **secure-boot-with-certificates**]
16. On a Get request, this property is returned only when the CPC has feature **secure-boot-with-certificates**.

Each nested boost-info object contains the following properties.

Table 522. boost-info nested object properties		
Name	**Type**	**Description**
boost-type	String Enum	The type of boost that is being used. Valid values are: • **"not-specified"** – the boost type is not provided by the operating system running on this LPAR. • **"not-active"** – indicates no boost is being used. • **"ipl"** – boost used during IPL of the operating system running in the LPAR for additional processing capacity. • **"shutdown"** – boost used during shutdown of the operating system running in the LPAR for additional processing capacity. • **"recovery-process-boost"** – boost used during system recovery events for additional processing capacity. Each individual recovery process boost is 5 minutes or less.
total-rpb-time	Integer	The total daily allowed recovery process boost time in seconds for the current LPAR.
remaining-rpb-time	Integer	The remaining allowed recovery process boost time in seconds for the current LPAR. This value will be decreased at the end of each recovery process boost. This value is reset to **total-rpb-time** value once daily.

Each nested central-storage-allocation object contains the following properties.

Table 523. central-storage-allocation nested object properties		
Name	**Type**	**Description**
origin	Long	The origin in megabytes of central storage in memory or null if the value is not available.
initial	Long	The initial amount of central storage in megabytes that is allocated or null if the value is not available.
current	Long	The current amount of central storage in megabytes that is allocated or null if the value is not available.
maximum	Long	The maximum amount of central storage in megabytes that is allocated or null if the value is not available.
gap	Long	The gap in megabytes from this partition's central storage to the start of the next partition's central storage, or null if the value is not available.
storage-element-type	String Enum	Type of storage element. Valid values are: • **"central"** - central storage • **"initial"** - initial expanded storage that is used as central storage • **"reserved"** - reserved expanded storage that is used as central storage.

Each nested expanded-storage-allocation object contains the following properties.

Table 524. expanded-storage-allocation nested object properties		
Name	**Type**	**Description**
origin	Long	The origin in megabytes of expanded storage in memory.
initial	Long	The initial amount of expanded storage in megabytes that is allocated.
current	Long	The current amount of expanded storage in megabytes that is allocated.
maximum	Long	The maximum amount of expanded storage in megabytes that is allocated.
gap	Long	The gap in megabytes from this partition's expanded storage to the start of the next partition's expanded storage.

List Logical Partitions of CPC

The `List Logical Partitions of CPC` operation lists the logical partitions of a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/logical-partitions
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC.

Query Parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`logical-partitions`	Array of logical-partition-info objects	Array of nested logical-partition-info objects (described in the next table)

Each nested logical-partition-info object contains the following fields:

Field name	Type	Description
`object-uri`	String/URI	Canonical URI path of the Logical Partition object
`name`	String	The name of the Logical Partition object
`status`	String Enum	The current status of the Logical Partition object

Field name	Type	Description
`request-origin`	Boolean	If true, the Logical Partition object is the same as the origin of the request. If false, the Logical Partition object is not the same as the origin of the request. **Note:** This property is only returned when the BCPii interface was used for the request.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Logical Partition objects that belong to a CPC. The object URI, display name, and status are provided for each.

If the **name** query parameter is specified, the returned list is limited to those Logical Partition objects that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

For the web services interface an object is only included in the list if the API user has object-access permission for that object. For the BCPii interface an object is only included in the list if the source partition has receive BCPii security controls permissions for the Logical Partition object.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1194.

If the CPC is in DPM mode, or there are no logical partitions defined to the CPC, or no logical partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Object-access permission to any Logical Partition object to be included in the result.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1194.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.

HTTP error status code	Reason code	Description
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/logical-partitions HTTP/1.1
x-api-session: 65aw2jahugn1wop51hsq0c6aldkkx773dz9ulirrvg2z853m4u
```

Figure 634. List Logical Partitions of CPC: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 16:58:36 GMT
content-type: application/json;charset=UTF-8
content-length: 374
{
   "logical-partitions": [
      {
         "name": "APIVM1",
         "object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
         "status": "operating"
      },
      {
         "name": "ZOS",
         "object-uri": "/api/logical-partitions/458e44e1-b0c2-391b-83ff-ecfd847295bd",
         "status": "not-operating"
      }
   ]
}
```

Figure 635. List Logical Partitions of CPC: Response

List Permitted Logical Partitions

The `List Permitted Logical Partitions` operation lists logical partitions to which the API user has object-access permission. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/console/operations/list-permitted-logical-partitions
```

Query parameters:

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **name** property.
`activation-mode`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **activation-mode** property. Value must be a valid logical partition **activation-mode** property value.

Name	Type	Rqd/Opt	Description
status	String Enum	Optional	Filter string to limit returned objects to those that have a matching **status** property. Value must be a valid logical partition **status** property value.
has-unacceptable-status	Boolean	Optional	Filter string to limit returned objects to those that have a matching **has-unacceptable-status** property. Valid values are **true** and **false**.
cpc-name	String	Optional	Filter pattern (regular expression) to limit returned objects to those whose parent CPC has a matching **name** property.
additional-properties	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties included in each **logical-partition-info**. This is a list of comma-separated strings where each string is a property name defined in the Logical Partition's data model. [Added by feature **secure-boot-with-certificates**]

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
logical-partitions	Array of logical-partition-info objects	Array of nested logical-partition-info objects as described in the next table.

Each nested logical-partition-info object contains the following fields:

Field name	Type	Description
name	String	The **name** property of the Logical Partition object.
object-uri	String/URI	The **object-uri** property of the Logical Partition object.
activation-mode	String Enum	The **activation-mode** property of the Logical Partition object.
status	String Enum	The **status** property of the Logical Partition object.
has-unacceptable-status	Boolean	The **has-unacceptable-status** property of the Logical Partition object.
cpc-name	String	The **name** property of the Logical Partition's parent CPC object.
cpc-object-uri	String/URI	The **object-uri** property of the Logical Partition's parent CPC object.
se-version	String	The **se-version** property of the Logical Partition's parent CPC object.

Field name	Type	Description
`request-origin`	Boolean	If true, the Logical Partition object is the same as the origin of the request. If false, the Logical Partition object is not the same as the origin of the request. **Note:** This property is only returned when the BCPii interface was used for the request.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing an operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

For the web services interface this operation lists the Logical Partition objects to which the API user has object-access permission. For the BCPii interface this operation lists the Logical Partition objects to which the source partition has receive BCPii security controls permissions. Some basic properties are provided for each logical partition that is included in the response.

If the **name** query parameter is specified, the returned list is limited to those logical partitions that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

If the **activation-mode** query parameter is specified, the parameter is validated to ensure it is a valid logical partition **activation-mode** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those logical partitions that have an **activation-mode** property matching the specified value. If the **activation-mode** parameter is omitted, no such filtering is performed.

If the **status** query parameter is specified, the parameter is validated to ensure it is a valid logical partition **status** property value. If the value is not valid, HTTP status code 400 (Bad Request) is returned. If the value is valid, the returned list is limited to those logical partitions that have a **status** property matching the specified value. If the **status** parameter is omitted, no such filtering is performed.

If the **has-unacceptable-status** query parameter is specified, the returned list is limited to those logical partitions that have a **has-unacceptable-status** property matching the specified value. If the **has-unacceptable-status** parameter is omitted, no such filtering is performed.

If the **cpc-name** query parameter is specified, the returned list is limited to those logical partitions whose parent CPC's **name** property matches the specified filter pattern. If the **cpc-name** parameter is omitted, no such filtering is performed.

For the web services interface a logical partition is included in the list only if the API user has object-access permission to that object. For the BCPii interface a logical partition is included in the list only if the source partition has receive BCPii security controls permissions. If there is a logical partition to which the API user does not have permission, that object is omitted from the list, but no error status code results.

If there are no logical partitions known or if no logical partitions are to be included in the response due to filtering or access permissions, an empty list is provided and the operation completes successfully.

If the **additional-properties** parameter is specified, additional properties are included in the returned list. The properties to be included is a list of comma-separated strings where each string is a property name defined in the Logical Partition's “Data model” on page 1169. If the **additional-properties** parameter is omitted, no such properties will be included. [Added by feature **secure-boot-with-certificates**]

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, object-access permission to the Logical Partition objects included in the response body.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for any Logical Partition object to be included in the result.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1197.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/console/operations/list-permitted-logical-partitions HTTP/1.1
x-api-session: 3y0qwkkon9m03c5o81oo2fz7ozezkinczcdwwr4gauzgogb5xq
```

Figure 636. List Permitted Logical Partitions: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 28 Aug 2017 18:07:25 GMT
content-type: application/json;charset=UTF-8
content-length: 860
{
   "logical-partitions":[
      {
         "activation-mode":"general",
         "cpc-name":"M87",
         "cpc-object-uri":"/api/cpcs/f2eb7e56-0c87-3646-887e-ec735b3844cd",
         "has-unacceptable-status":false,
         "name":"S50",
         "object-uri":"/api/logical-partitions/d39347d9-855a-3199-9ef6-a1701b7b17b4",
         "se-version":"2.14.0",
         "status":"operating"
      },
      {
         "activation-mode":"esa390",
         "cpc-name":"S15",
         "cpc-object-uri":"/api/cpcs/f6f629ca-f2c5-3f71-a80f-d9b91a492549",
         "has-unacceptable-status":false,
         "name":"APIVM2",
         "object-uri":"/api/logical-partitions/c53c7aa7-444b-3f05-87b4-fb94802240b8",
         "se-version":"2.13.1",
         "status":"operating"
      },
      {
         "activation-mode":"not-set",
         "cpc-name":"S15",
         "cpc-object-uri":"/api/cpcs/f6f629ca-f2c5-3f71-a80f-d9b91a492549",
         "has-unacceptable-status":true,
         "name":"ZOS",
         "object-uri":"/api/logical-partitions/17c4bc30-96b5-327e-8b7d-59e1b4b7261e",
         "se-version":"2.13.1",
         "status":"not-activated"
      }
   ]
}
```

Figure 637. List Permitted Logical Partitions: Response

Usage note

The response body of this operation is similar to that of the `Get Inventory` operation, but it returns only a subset of logical partition properties. The response also includes some properties of the parent CPC, regardless of whether the API user has object-access permission to that CPC.

Get Logical Partition Properties

The `Get Logical Partition Properties` operation retrieves the properties of a single Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/logical-partitions/{logical-partition-id}
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Query parameters:

Name	Type	Rqd/Opt	Description
properties	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the Logical Partition object's data model.
cached-acceptable	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**.
group-uri	String/ URI	Optional	The object URI for the Custom Group object to be used for determining the value of the **next-activation-profile-name** property. If omitted the system-defined Logical Partition group will be used.

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body provides the current values of the properties for the Logical Partition object as defined in "Data model" on page 1169.

Description

The URI path must designate an existing Logical Partition object and the API user must have access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some logical partition property values are periodically fetched from the Support Element and cached for quick access by the APIs. Due to the nature of this caching support, the cached value of a property may differ from the actual value at any point in time. While the cache is kept reasonably current, there are no guarantees about the latency of the cache, nor is there any latency information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined in "Data model" on page 1169.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the Logical Partition object designated by *{logical-partition-id}*.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1201.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9 HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 638. Get Logical Partition Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Tue, 05 Mar 2019 17:16:16 GMT
content-type: application/json;charset=UTF-8
content-length: 3444
{
   "absolute-aap-capping":{"type": "none"},
   "absolute-cf-capping":{"value": 88.52, "type": "processors"},
   "absolute-ifl-capping":{"type": "none"}
   "absolute-processing-capping":{"value": 0.01, "type": "processors"},
   "absolute-ziip-capping":{"value": 2.01, "type": "processors"},
   "acceptable-status": [
      "operating"
   ],
   "activation-mode": "esa390",
   "additional-status": "",
   "class": "logical-partition",
   "cluster-name": "",
   "current-aap-processing-weight": null,
   "current-aap-processing-weight-capped": null,
   "current-cf-processing-weight": null,
   "current-cf-processing-weight-capped": null,
   "current-ifl-processing-weight": null,
   "current-ifl-processing-weight-capped": null,
   "current-processing-weight": 100,
   "current-processing-weight-capped": false,
   "current-vfm-storage": 256,
   "current-ziip-processing-weight": null,
   "current-ziip-processing-weight-capped": null,
   "defined-capacity": 0,
   "description": "LPAR Image",
   "group-profile-capacity": null,
   "group-profile-uri": null,
   "has-operating-system-messages": false,
   "has-unacceptable-status": false,
   "initial-aap-processing-weight": null,
   "initial-aap-processing-weight-capped": null,
   "initial-cf-processing-weight": null,
   "initial-cf-processing-weight-capped": null,
   "initial-ifl-processing-weight": null,
   "initial-ifl-processing-weight-capped": null,
   "initial-processing-weight": 100,
   "initial-processing-weight-capped": false,
   "initial-vfm-storage": 64,
   "initial-ziip-processing-weight": null,
   "initial-ziip-processing-weight-capped": null,
   "is-locked": false,
   "is-sub-capacity-boost-active":false,
   "is-ziip-capacity-boost-active":false,
   "last-used-activation-profile": "APIVM1",
   "last-used-boot-record-logical-block-address":"C8",
   "last-used-disk-partition-id":0,
   "last-used-load-address": "05402",
   "last-used-load-parameter": "TESTMODE",
   "last-used-logical-unit-number":"0015000000000000",
   "last-used-operating-system-specific-load-parameters":"0002,0,fa163e057ee54",
   "last-used-secure-boot": false,
   "last-used-world-wide-port-name":"50017380EB0B0142",
   "maximum-aap-processing-weight": null,
   "maximum-cf-processing-weight": null,
   "maximum-ifl-processing-weight": null,
```

Figure 639. Get Logical Partition Properties: Response (Part 1)

```
    "maximum-processing-weight": 200,
    "maximum-vfm-storage": 512,
    "maximum-ziip-processing-weight": null,
    "minimum-aap-processing-weight": null,
    "minimum-cf-processing-weight": null,
    "minimum-ifl-processing-weight": null,
    "minimum-processing-weight": 50,
    "minimum-ziip-processing-weight": null,
    "name": "APIVM1",
    "next-activation-profile-name": "APIVM1",
    "number-general-purpose-cores": 5,
    "number-general-purpose-processors": 5,
    "number-icf-cores": 0,
    "number-icf-processors": 0,
    "number-ifl-cores": 0,
    "number-ifl-processors": 0,
    "number-reserved-general-purpose-cores": 2,
    "number-reserved-general-purpose-processors": 2,
    "number-reserved-icf-cores": 0,
    "number-reserved-icf-processors": 0,
    "number-reserved-ifl-cores": 0,
    "number-reserved-ifl-processors": 0,
    "number-reserved-ziip-cores": 1,
    "number-reserved-ziip-processors": 1,
    "number-ziip-cores": 1,
    "number-ziip-processors": 1,
    "object-id": "c7eb8134-826e-3a71-8d1a-00d706c874e9",
    "object-uri": "/api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9",
    "os-ipl-token": "0000000000000000",
    "os-level": "6.2.0",
    "os-name": "APIVM1",
    "os-type": "z/VM",
    "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
    "partition-number": 1,
    "processor-usage":"shared",
    "program-status-word-information": [
       {
          "cpid": "00",
          "psw": "0706400080000000000000000000000"
       },
       {
          "cpid": "01",
          "psw": "0706400080000000000000000000000"
       }
    ],
    "speed-boost":{
       "boost-type":"recovery-process-boost",
       "total-rpb-time":"1800",
       "remaining-rpb-time":"1800"
    },
    "status": "operating",
    "storage-central-allocation": [
       {
           "origin": 16697344,
           "initial": 1024,
           "current": 1024,
           "maximum": 1024,
           "gap": 1024,
           "storage-element-type": "central"
       },
       {
           "origin": 16696832,
           "initial": null,
           "current": 512,
           "maximum": null,
           "gap": 0,
           "storage-element-type": "initial"
       }
    ],
```


Figure 640. Get Logical Partition Properties: Response (Part 2)

```
    "storage-expanded-allocation": [
       {
           "origin": 16776704,
           "initial": 512,
           "current": 512,
           "maximum": 512,
           "gap": 0
       }
    ],
    "sysplex-name": "SSICAPI1",
    "workload-manager-enabled": true
    "ziip-boost":{
       "boost-type":"recovery-process-boost",
       "total-rpb-time":"1800",
       "remaining-rpb-time":"1800"
    }
 }
```

Figure 641. Get Logical Partition Properties: Response (Part 3)

Update Logical Partition Properties

The `Update Logical Partition Properties` operation updates one or more writable properties of the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Query parameters:

Name	Type	Rqd/Opt	Description
`group-uri`	String/ URI	Optional	The object URI for the Custom Group object to be used for updating the value of the **next-activation-profile-name** property. If omitted the system-defined Logical Partition group will be used.

Request body contents

The request body is expected to contain one or more field names representing writable logical partition properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Logical Partition object type to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the object is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- If any of the **workload-manager-enabled**, **defined-capacity**, or **absolute-*-capping** properties is to be updated, object-access permission to the logical partition's parent CPC object.
- Action/task permission to the **Image Details** task, with the following exceptions:
  - If the **next-activation-profile-name** property is to be updated, action/task permission for the **Change Object Options** task, the **Customize/Delete Activation Profiles** task, or the **Image Details** task.
  - If any of the **number-*-cores** or **number-reserved-*-cores** properties is to be updated, action/task permission for the **Logical Processor Add** task or the **Image Details** task.
  - If any of the **workload-manager-enabled**, **defined-capacity**, ***-processing-weight**, **initial-*-processing-weight-capped**, or **absolute-*-capping** properties is to be updated, either of the following:
    - Action/task permission for the **Change LPAR Controls** task.
    - Action/task permission for the **Image Details** task but not the **Change LPAR Controls** task in view-only mode.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The API user does not have the required permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	0	A general conflict was detected during common request validation. See the returned message field for details
	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Activate Logical Partition

The `Activate Logical Partition` operation activates and optionally loads the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/activate
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`activation-profile-name`	String (1-16)	Optional when the **load-type** is **"none"** Invalid otherwise	The name of the activation profile to be used for the request. If this field is not provided and: • **load-type** is **"none"** or not specified, the request uses the profile name specified in the **next-activation-profile-name** property of the target Logical Partition object. • **load-type** is not **"none"**, the request first activates the Logical Partition using the image activation profile which has the same name as the Logical Partition, then it performs the load using the load parameters specified in this request body. If this field is provided, then the **load-type** cannot be specified or must be specified to **"none"**. [Updated by feature **secure-boot-with-certificates**]
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.
`load-type`[1]	String Enum	Optional	The load type to be used after the activation. One of the following: • **"none"** - No load operation is performed. • **"ipltype-standard"** - A Channel Command Work (CCW) standard load operation is performed. This is present for compatibility, and the more specific **"ipltype-tape-load"** or **"ipltype-eckd-ccw-load"** value is preferred. If this value is specified, and the **secure-boot-with-certificates** feature is available on the CPC, the load is completed with a **load-type** value of **"ipltype-eckd-ccw-load"**. The load operation will behave the same as **"ipltype-standard"** though the **last-used-load-type** on the Logical Partition object will reflect the changed value. • **"ipltype-scsi"** - A list-directedSCSI load operation is performed. • **"ipltype-nvmeload"** - A list-directedNVMe load operation is performed. • **"ipltype-tape-load"**[2] - A Channel Command Word (CCW) tape OS load is performed. • **"ipltype-eckd-ccw-load"**[2] - A Channel Command Word (CCW) ECKD OS load is performed. • **"ipltype-eckd-ld-load"**[2] - A list-directed ECKD OS load is performed. When **load-type** is not **"none"**, the fields that are required for the load type must be specified, and fields that are not applicable to the load type are ignored. Default: **"none"** [Updated by feature **secure-boot-with-certificates**]

Field name	Type	Rqd/ Opt	Description
`load-address`[1]	String (1-5)	Required when **load-type** is **"ipltype-scsi"**, **"ipltype-nvmeload"**, or **"ipltype-eckd-ld-load"** Invalid when the **load-type** is **"none"** Optional otherwise	The hexadecimal address of an I/O device that provides access to the control program to be loaded. When **load-type** is **"ipltype-nvmeload"**, this field shall be a 4-character function ID (FID). If the input value is less than 4 in length, it will be right-justified and padded with zeros to 4 characters. The final value must be in the range "0000" to "FFFF". When **load-address** is not supplied and **load-type** is **"ipltype-standard"**, **"ipltype-eckd-ccw-load"**, or **"ipltype-tape-load"**, the operation will try to load using the address used in the last load. Otherwise, the input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". [Updated by feature **secure-boot-with-certificates**]
`load-parameter`[1]	String (0-8)	Optional	Some control programs support the use of this field to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A- Z, @, $, #, blank and period. Default: an empty string
`secure-boot`[1]	Boolean	Optional when the **load-type** is **"ipltype-scsi"**, **"ipltype-nvmeload"**, or **"ipltype-eckd-ld-load"** Invalid when the **load-type** is **"none"**, **"ipltype-standard"**, **"ipltype-tape-load"**, or **"ipltype-eckd-ccw-load"**	If **true**, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the logical partition. The load will fail if the signatures do not match. Default: **false** [Updated by feature **secure-boot-with-certificates**]
`timeout`[1]	Integer (60-600)	Optional when the **load-type** is **"ipltype-standard"**, **"ipltype-eckd-ccw-load"**, or **"ipltype-tape-load"** Invalid otherwise	Amount of time, in seconds, to wait for the load to complete. Default: 60 [Updated by feature **secure-boot-with-certificates**]
`world-wide-port-name`[1]	String (1-16)	Required when **load-type** is **"ipltype-scsi"** Invalid otherwise	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal. [Updated by feature **secure-boot-with-certificates**]
`logical-unit-number`[1]	String (1-16)	Required if **load-type** is **"ipltype-scsi"** Invalid otherwise	The hexadecimal logical unit number (LUN) to be used for the SCSI load. [Updated by feature **secure-boot-with-certificates**]
`disk-partition-id`[1]	Integer (0-30)	Optional when the **load-type** is **"ipltype-nvmeload"**, **"ipltype-scsi"**, or **"ipltype-eckd-ld-load"**. Invalid otherwise	The **disk-partition-id** (also called the boot program selector) to be used for the list-directed load. This value cannot be specified if **disk-partition-id-automatic** is **true**. Default: 0 [Updated by feature **secure-boot-with-certificates**]

Field name	Type	Rqd/ Opt	Description
`disk-partition-id-automatic`[2]	Boolean	Optional when the **load-type** is **"ipltype-nvmeload"**, **"ipltype-scsi"**, or **"ipltype-eckd-ld-load"**. Invalid otherwise	Whether the **disk-partition-id** should be determined automatically. This field cannot be specified if **disk-partition-id** is specified. Default: false [Added by feature **secure-boot-with-certificates**]
`operating-system-specific-load-parameters`[1]	String (0-256)	Optional when the **load-type** is **"ipltype-nvmeload"**, **"ipltype-scsi"**, or **"ipltype-eckd-ld-load"**. Invalid otherwise	The operating system specific load parameters to be used for the list-directed load operation. Default: an empty string [Updated by feature **secure-boot-with-certificates**]
`boot record-logical-block-address`[1]	String (1-16)	Optional when the **load-type** is **"ipltype-nvmeload"**, or **"ipltype-scsi"**. Invalid otherwise	The hexadecimal boot record logical block address to be used for the SCSI or NVMe load. Default: hex zeroes [Updated by feature **secure-boot-with-certificates**]
`os-ipl-token`[1]	String (1-16)	Optional when the **load-type** is **"ipltype-standard"**, **"ipltype-tape-load"**, **"ipltype-scsi"**, or **"ipltype-eckd-ld-load**. Invalid otherwise	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** string was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A- Z), left-justified. [Updated by feature **secure-boot-with-certificates**]
`boot-record-location-use-volume-label`[2]	Boolean	Optional when **load-type** is **"ipltype-eckd-ld-load"**. Invalid otherwise	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** values should be determined by the volume label. This value cannot be set to true if any of **boot-record-location-cylinder**, **boot-record-location-head** or **boot-record-location-record** are specified. Default: true if none of the above values is specified. Otherwise, false. [Added by feature **secure-boot-with-certificates**]
`boot-record-location-cylinder`[2]	String (1-7)	Optional when **load-type** is **"ipltype-eckd-ld-load"**. Invalid otherwise	The boot record location cylinder value in hexadecimal. This may not be specified if **boot-record-location-use-volume-label** is **true**. Default: "0" [Added by feature **secure-boot-with-certificates**]
`boot-record-location-head`[2]	String (1)	Optional when **load-type** is **"ipltype-eckd-ld-load"**. Invalid otherwise	The boot record location head value in hexadecimal. This may not be specified if **boot-record-location-use-volume-label** is **true**. Default: "0" [Added by feature **secure-boot-with-certificates**]
`boot-record-location-record`[2]	String (1-2)	Optional when **load-type** is **"ipltype-eckd-ld-load"**. Invalid otherwise	The boot record location record value in hexadecimal. The record may not be set to "0" or "00". This may not be specified if **boot-record-location-use-volume-label** is **true**. Default: "1" [Added by feature **secure-boot-with-certificates**]

Field name	Type	Rqd/ Opt	Description
Notes: [1] This field can be included in the request body only when the associated SE version is 2.16.0 or later. [2]This field or Enum value can be included in the request body only when the CPC has feature **secure-boot-with-certificates**.			

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1211. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Activation is a process that makes a logical partition operational, which means either:

- The logical partition is ready to have a control program or operating system loaded, or
- The logical partition has loaded and is running a control program or operating system.

Activating a logical partition includes:

- Initializing the logical partition
- Allocating system resources to the logical partition
- Loading the logical partition with a control program or operating system.

Since the status of the logical partition determines which operations must be performed during activation to make the logical partition operational, one or more operations listed above may not be performed during activation.

Performing the Load Logical Partition operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1211.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Activate** task.
  - Action/task permission for the **Load** task, if a load after the activation is requested.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1210.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	15	The request body contains an activation profile name and a **load-type** that is not **"none"**, or a load parameter is specified and it is not applicable to the specified **load-type**.
	20	The request body contains a load parameter and the associated SE version is 2.15.0 or earlier.
	264	The specified IPL Token value does not match the current IPL Token value.
	306	Loading the Logical Partition object is not valid in the current activation mode.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.

HTTP error status code	Reason code	Description
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Deactivate Logical Partition

The `Deactivate Logical Partition` operation deactivates the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/deactivate
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1213. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Deactivation is an orderly process for terminating a logical partition.

Deactivating a logical partition includes:

- Unloading the logical partition's control program or operating system
- Freeing system resources allocated to the logical partition.

After the logical partition is deactivated, the logical partition is no longer operational

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1213.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Deactivate** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1212.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.

HTTP error status code	Reason code	Description
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Reset Normal

The `Reset Normal` operation initializes a system or logical partition by clearing its pending interruptions, resetting its channel subsystem and resetting its processors. A reset prepares a system or logical partition for loading it with an operating system. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/reset-normal
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.
`os-ipl-token`	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1216. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1216.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Reset Normal** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1214.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Reset Clear

The `Reset Clear` operation initializes system or logical partition by clearing its pending interruptions, resetting its channel subsystem and resetting its processors. A reset prepares a system or logical partition for loading it with an operating system and clears main memory of the system or logical partition. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/reset-clear
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.
`os-ipl-token`	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9,A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1218. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1218.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Reset Clear** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1216.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Load

The Load operation is a single, flexible, asynchronous load operation that allows you to load an operating system or a dump program from any available device. It includes all functionality of the existing `Load Logical Partition`, `SCSI Load`, `SCSI Dump`, `NVMe Load`, and `NVMe Dump` operations. Those operations will not be updated with new functionality for future load changes. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/load-program
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields.

Note: Read the **Required/Optional/Invalid** column carefully to determine which fields are required and optional given other provided parameters

Field name	Type	Rqd/ Opt/ Invalid	Description
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (**true**) or not (**false**). Default: false

Field name	Type	Rqd/ Opt/ Invalid	Description
`device-type`	String Enum	Required	The type of the device which contains the program to be loaded. One of: • **"tape"** - A tape device. This is a **device-type** that may be referred to elsewhere as a "standard" load. Tape devices always use a **"ccw" load-operation-type**. • **"eckd"** - An ECKD device. When the **"ccw" load-operation-type** is specified, this also may be referred to elsewhere as a "standard" load. • **"nvme"** - An NVMe device. NVMe devices always use a **"list-directed" load-operation-type**. • **"scsi"** - A SCSI device. SCSI devices always use a **"list-directed" load-operation-type**.
`load-program-type`	String Enum	Required	The type of program to be loaded. One of: • **"os"** - An operating system • **"dump"** - A dump program
`load-operation-type`	String Enum	Required for **device-type "eckd"** Optional for all others	The method used to complete the load. One of: • **"ccw"** - Channel Command Word load • **"list-directed"** - A List Directed load A **device-type "eckd"** load may be either List Directed or Channel Command Word, so the value must be specified. Default: No default for **device-type "eckd"**. For **device-type "tape"**, the default is **"ccw"**. For **device-type "nvme"** and **"scsi"**, the default is **"list-directed"**.
`load-address`	String (1-5)	Optional when **load-operation-type** is **"ccw"** Required otherwise	The hexadecimal address of an I/O device that provides access to the control program to be loaded. When **device-type** is **"nvme"** this field shall be a 4-character function ID (FID). If the input value is less than 4 in length, it will be right justified and padded with zeros to 4 characters. The final value must be in the range "0000" to "FFFF". Otherwise, the input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". Default: When the **load-operation-type** is **"ccw"**, the operation will load using the logical-partition's last-used-load-address. Otherwise, "0".
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this string is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, blank and period. Default: An empty string

Field name	Type	Rqd/ Opt/ Invalid	Description
`secure-boot`	Boolean	Optional when **load-operation-type** is **"list-directed"** Invalid otherwise	If true, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the logical partition. The load will fail if the signatures do not match. Default: false
`clear-memory`	Boolean	Optional when **load-program-type** is **"os"** Invalid otherwise	Whether the main memory should be cleared before loading an operating system. Default: false
`store-status`	Boolean	Optional when **device-type** is **"eckd"** or **"tape"** and the **load-program-type** is **"dump"** Invalid otherwise	This field indicates whether the store status function should be invoked before performing the dump (**true**) or not (**false**). The store status function stores the current values of the processing unit timer, the clock comparator, the program status word, and the contents of the processor registers in their assigned absolute storage locations. Default: false
`timeout`	Integer (60-600)	Optional when **load-operation-type** is **"ccw"** Invalid otherwise	Amount of time, in seconds, to wait for the load to complete before failing asynchronously. Default: 60
`world-wide-port-name`	String (1-16)	Required when **device-type** is **"scsi"** Invalid otherwise	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal.

Field name	Type	Rqd/ Opt/ Invalid	Description
`logical-unit-number`	String (1-16)	Required when **device-type** is **"scsi"** Invalid otherwise	The hexadecimal logical unit number (LUN) to be used for the SCSI Load.
`boot-program-selector`	Integer (0-30)	Optional when **load-operation-type** is **"list-directed"** Invalid otherwise	The boot program selector (also known as the disk partition id) to be used for the list-directed load. This value cannot be specified if the **boot-program-selector-automatic** value is specified and set to **true**.
`boot-program-selector-automatic`	Boolean	Optional when **load-operation-type** is **"list-directed"** Invalid otherwise	Whether the boot-program-selector value should be determined automatically. This value cannot be set to true if **boot-program-selector** is specified. Default: True if a **boot-program-selector** value is not specified in the request.
`boot-record-lba`	String (1-16)	Optional for **device-type "scsi"** or **"nvme"** Invalid otherwise	The hexadecimal boot record logical block address to be used for the load.
`os-load-parameters`	String (0-256)	Optional when **load-operation-type** is **"list-directed"** Invalid otherwise	The operating system specific load parameters to be used for the load operation. Default: an empty string

Field name	Type	Rqd/ Opt/ Invalid	Description
`boot-record-location-cylinder`	String (1-7)	Optional when **device-type** is **"eckd"** and **load-operation-type** is **"list-directed"** Invalid otherwise	The boot record location cylinder value in hexadecimal. This may not be specified if **boot-record-location-use-volume-label** is true. Default: "0"
`boot-record-location-head`	String (1)	Optional when **device-type** is **"eckd"** and **load-operation-type** is **"list-directed"** Invalid otherwise	The boot record location head value in hexadecimal. This may not be specified if **boot-record-location-use-volume-label** is true Default: "0"
`boot-record-location-record`	String (1-2)	Optional when **device-type** is **"eckd"** and **load-operation-type** is **"list-directed"** Invalid otherwise	The boot record location record value in hexadecimal. The record may not be specified as "0" or "00". This may not be specified if **boot-record-location-use-volume-label** is true Default: "1"
`boot-record-location-use-volume-label`	Boolean	Optional when **device-type** is **"eckd"** and **load-operation-type** is **"list-directed"** Invalid otherwise	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** values should be determined by the volume label. This value cannot be set to **true** if any of **boot-record-location-cylinder**, **boot-record-location-head** or **boot-record-location-record** are specified. Default: True if none of the above values is specified

Field name	Type	Rqd/ Opt/ Invalid	Description
`os-ipl-token`	String (1-16)	Optional when **device-type** is **"tape"**, **"eckd"**, or **"scsi"** Invalid otherwise	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** field was retrieved. It is recommended that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left-justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in . The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1224.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1223.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	375	The request body contains a value that is incompatible with the **device-type**, **load-program-type** or **load-operation-type**.
	376	The request body is missing a value required for a **device-type** or **load-operation-type**.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The API user does not have action/task permission for the **Load** task
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
	4	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object where the owning CPC's SE version is 2.16.0 with the suitable MCL bundle or later.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Load Logical Partition

The `Load Logical Partition` operation resets a logical partition, to prepare it for loading an operating system, and loads the operating system. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/load
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields. If none of the optional fields are included, an empty request body must be supplied.

Field name	Type	Rqd/ Opt	Description
`load-address`	String (1-5)	Optional	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". When **load-address** is not supplied, the operation will try to load using the address used in the last load.
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, $, #) are also allowed when the **se-version** property of the associated CPC is **"2.14.0"** or later.
`clear-indicator`	Boolean	Optional	Whether memory should be cleared before performing the Load (true) or not cleared (false). The default value is true.
`timeout`	Integer (60-600)	Optional	Amount of time, in seconds, to wait for the Load to complete. The default timeout value is 60 seconds.
`store-status-indicator`	Boolean	Optional	Whether status should be stored before performing the Load (true) or not stored (false). The default is false.
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.
`os-ipl-token`	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1227. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Beginning with API version 4.10, if **store-status-indicator** is specified as **true**, the **last-used-load-type property** of the target Logical Partition will be set to **"ipltype-ccw-dump"**; otherwise, it will be set to **"ipltype-ccw-load"**. [Updated by feature **secure-boot-with-certificates**]

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1227.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1225.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	264	The specified IPL Token value does not match the current IPL Token value.
	306	This operation is not valid in the current activation mode.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed or was rejected due to the current logical partition status and use of the force=false parameter. If rejected due to force=false, the logical partition status is unchanged. If the operation failed, the logical partition status is unknown. Refer to the message parameter in the error response body for details.

Usage note

Beginning with API version 4.10 this operation will not be enhanced to support new load functionality. Instead, API users are encouraged to use "Load" on page 1218, which includes the functionality of this operation as well as additional capabilities. [Updated by feature **secure-boot-with-certificates**]

Load Logical Partition from FTP

The `Load Logical Partition from FTP` operation loads system software or utility programs from an FTP server.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/load-from-ftp
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields.

Field name	Type	Rqd/ Opt	Description
`host-name`	String/ Hostname, String/ IPv4 Address, String/ IPv6 Address	Required	Specify the host name or IP address of the FTP server.
`user-name`	String	Required	Specify the user name for the account on the target FTP server.
`password`	String	Required	Specify the password that is associated with the user name on the target FTP server.
`protocol`	String Enum	Optional	Chose a secure network protocol for transferring files. See the usage notes for more information. • **"ftp"** - File Transfer Protocol • **"ftps"** - FTP Secure • **"sftp"** - SSH File Transfer Protocol Default: **"ftp"**
`file-path`	String	Required	The path to the file to be read from the FTP server and loaded into the logical partition.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1229. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

This operation asynchronously loads system software or utility programs from an FTP server. When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1229.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Load from Removable Media or Server** task.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1228.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The API user does not have action/task permission for the **Load from Removable Media or Server** task.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	306	This operation is not valid in the current activation mode.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.

HTTP error status code	Reason code	Description
409 (Conflict)	342	Unable to connect to the FTP server
	343	The FTP/SFTP/FTPS login failed, because the combination of user name and password was not accepted.
	344	The SFTP connection failed because a host key was not retrieved for the specified host
	345	The FTPS connection failed because the connection attempt to the host has been refused.
	346	The FTPS connection failed because a timeout occurred attempting to connect to the FTPS server.
	347	The specified file was not found.
500 (Server Error)	263	Operation failed.

Usage notes:

To use FTP Secure protocol, use the **Certificate Management** task to import an FTPS server certificate. From the **Advanced** drop-down, select **Manage Trusted Signing Certificate**. The Manage Trusted Signing Certificate window is displayed. From the **Import** drop-down, select **From Remote Server**. The Import Remote Certificate window is displayed. Provide the IP/Host address and port number, then click **OK** to confirm. The Confirm Import window is displayed, then click **Yes** after you verified the certificate information.

To use SSH File Transfer Protocol, use the **Manage SSH Keys** task to import SSH server keys. Provide the SFTP server ID in the **Address** input area, then click **Add**.

PSW Restart

The `PSW Restart` operation restarts the first available processor of the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/psw-restart
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job

completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1232. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

Information about PSW Restart can be found on console help system.

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1232.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **PSW Restart** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1230.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Start Logical Partition

The `Start Logical Partition` operation starts the processors to process instructions of the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/start
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1233. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1233.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Start** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1232.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Stop Logical Partition

The `Stop Logical Partition` operation stops the processors from processing instructions of the Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/stop
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1235. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

This operation is not permitted for a logical partition whose **activation-mode** property is **"zaware"** or **"ssc"**.

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1235.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Stop** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1234.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	306	This operation is not valid in the current activation mode.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Send OS Command

The `Send OS Command` operation sends a command to the operating system running in a logical partition. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/send-os-cmd
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`is-priority`	Boolean	Optional	An indication of whether this is a priority operating system command. Set to **true** for priority operating system commands or **false** for non-priority operating system commands. The default is **false**.
`operating-system-command-text`	String (1-200)	Required	The text of the operating system command.

Description

This operation sends a command to the operating system running in the Logical Partition targeted by the request URI.

The URI path must designate an existing Logical Partition object and the API user must have access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. For the web services interface, in addition to having object-access permission to the Logical Partition, the API user must also have permission to the **Operating System Messages** task, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the Logical Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

On successful execution, the command is sent to the operating system running in the target Logical Partition object and status code 204 (No Content) is returned without supplying a response body.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Operating System Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designated an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	332	The messages interface is not available.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9/operations/send-os-cmd
  HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 69
{
  “is-priority”: false,
  “operating-system-command-text”: “help”
}
```

Figure 642. Send OS Command: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Mon, 01 Feb 2016 09:57:18 GMT

<No response body>
```

Figure 643. Send OS Command: Response

Open OS Message Channel

The `Open OS Message Channel` operation opens a message channel to the operating system running in a logical partition for a client of the JMS notification facility (see the "JMS basics" section of Chapter 4 for more information). SSE clients should refer to the Server-Sent Events Stream operations in Chapter 7 for asynchronous OS message support.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/open-os-message-channel
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

An optional request body can be specified as a JSON object with the following field:

Field name	Type	Rqd/Opt	Description
`include-refresh-messages`	Boolean	Optional	An indication of whether refresh operating system messages should be sent. Set to **true** to receive refresh messages, or **false** to prevent refresh messages. The default is **true**.

Response body contents

On successful completion, the response body contains a JSON object with the following field:

Field name	Type	Description
`topic-name`	String (1-128)	The name of the os-message-notification topic.

Description

This operation opens a message channel to the operating system running in the Logical Partition targeted by the request URI. The message channel is implemented as a JMS topic, specifically as an os-message-notification topic. See Chapter 4, "Asynchronous notification," on page 77 for information on JMS usage on the HMC. The API user can connect to this topic to start the flow of new (and refreshed) operating system messages.

The URI path must designate an existing Logical Partition object and the API user must have object-access permission to it. If either of these conditions is not met, status code 404 (Not Found) is returned. In addition to having object-access permission to the Logical Partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the Logical Partition is not active, there is no operating system running in the partition, or when the operating system is not enabled for console integration.

If an os-message-notification topic already exists for this logical partition for the current API session, the operation fails.

On successful execution, the message channel is opened and the os-message-notification topic name is returned in the response body.

Authorization requirements

This operation has the following authorization requirements:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
- Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1237.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

Table 525. Open OS Message Channel: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	331	An os-message-notification topic already exists for this logical partition for the current API session. Use the `Get Notification Topics` operation to determine the topic name.
	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/c7eb8134-826e-3a71-8d1a-00d706c874e9/operations/
   open-os-message-channel HTTP/1.1
x-api-session: 2ltfe2c2q3ti2b2pwq1wfwuzifoi4qymqa8ktzjep7dbyrll0k
content-type: application/json
content-length: 37
{ “include-refresh-messages”: true }
```

Figure 644. Open OS Message Channel: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Sat, 14 Sept 2013 18:03:00 GMT
content-type: application/json;charset=UTF-8
{ “topic-name”: “mikeuser.1osmsg.cpc1.lpar1” }
```

Figure 645. Open OS Message Channel: Response

List OS Messages of a Logical Partition

The `List OS Messages of a Logical Partition` operation lists all currently available operating system (OS) messages for a logical partition. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/logical-partitions/{logical-partition-id}/operations/list-os-messages
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target logical partition.

Query parameters:

Name	Type	Rqd/ Opt	Description
`begin-sequence-number`	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number less than this are omitted from the results. If not specified, then no such filtering is performed.
`end-sequence-number`	Long	Optional	A message sequence number to limit returned messages. OS messages with a sequence number greater than this are omitted from the results. If not specified, then no such filtering is performed.
`is-held`	Boolean	Optional	A Boolean value used to limit the returned messages to held or non-held messages. A value of true will result in only held messages being returned, while a value of false will result in only non-held messages being returned.
`is-priority`	Boolean	Optional	A Boolean value used to limit the returned messages to priority or non-priority messages. A value of true will result in only priority messages being returned, while a value of false will result in only non-priority messages being returned.

Name	Type	Rqd/ Opt	Description
`max-messages`	Integer	Optional	An integer value greater than zero that indicates the maximum number of messages to be returned. If specified, this query parameter can only be specified once. Use of this query parameter allows for the data returned to be limited. Using the sequence-number of the last message returned as the **begin-sequence-number** value on a subsequent invocation of this operation can get the next set of messages. **Note:** For operations using the BCPii interface this query parameter is required and cannot be a value greater than 100.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`os-messages`	Array of os-message-info objects	Array of nested os-message-info objects as described in the next table. The array elements are in order of increasing sequence number, unless that number has wrapped back around to 0 in which case the element with sequence number 0 follows the element with the highest sequence number, thus maintaining the temporal order of the messages.

Each nested os-message-info object contains the following fields:

Field name	Type	Description
`sequence-number`	Long	The sequence number assigned to this operating system message by the HMC. Although sequence numbers may wrap over time, this number can be considered a unique identifier for the message. It can be used for filtering purposes.
`message-text`	String	The text of the operating system message.
`message-id`	Long	The message identifier of the operating system message.
`timestamp`	Timestamp	The time when the operating system message was created or -1 if this information is not available from the corresponding operating system.
`sound-alarm`	Boolean	Specifies whether the operating system message should cause the alarm to be sounded (true) or not (false).
`is-priority`	Boolean	Specifies whether the operating system message is a priority message (true) or not (false). A priority message indicates a critical condition that requires immediate attention.
`is-held`	Boolean	Specifies whether the operating system message is a held message (true) or not (false). A held message is one that requires a response.
`prompt-text`	String	Specifies the prompt text that is associated with this operating system message or null indicating that there is no prompt text for this operating system message. The prompt text is used when responding to a message. The response is to be sent as an operating system command where the command is prefixed with the prompt text and followed by the response to the message.

Field name	Type	Description
`os-name`	String	Specifies the name of the operating system that generated this operating system message or null indicating there is no operating system name associated with this operating system message. This name is determined by the operating system itself and may be unrelated to the name of the logical partition in which the operating system is running.

Description

This operation lists the currently available messages from the operating system running in the specified logical partition. Only a certain amount of OS message data from each logical partition is preserved for retrieval by this operation. If the OS produces more than that amount, the oldest non-held, non-priority OS messages are no longer available. A gap in the sequence numbers indicates a loss of messages. A loss may be due to that space limitation, or it may be due to the deletion of messages by a console user or the OS.

If the request URI does not identify a Logical Partition object to which the API user has access permission, HTTP status code 404 (Not Found) is returned. For the web services interface, in addition to having object-access permission to the logical partition, the API user must also have permission to the **Operating System Messages** task or the **Operating System Messages** task in view-only mode, otherwise status code 403 (Forbidden) is returned. Status code 409 (Conflict) is returned when the message interface for the operating system running in the target logical partition is not available. Some examples are when the logical partition is not active, there is no operating system running in the logical partition, or when the operating system is not enabled for console integration.

If the **begin-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** less than that are omitted from the response. If the **end-sequence-number** query parameter is specified, then any OS messages with a **sequence-number** greater than that are omitted from the response. If the **is-held** query parameter is specified as true, then non-held messages are omitted. If the **is-held** query parameter is specified as false, then held messages are omitted. If the **is-priority** query parameter is specified as true, then non-priority messages are omitted. If the **is-priority** query parameter is specified as false, then priority messages are omitted. If the **max-messages** query parameter is specified, then the number of returned messages will not exceed this value.

If there are no available OS messages for the specified logical partition or if no OS messages are to be included in the response due to filtering, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Operating System Messages** task, or the **Operating System Messages** task in view-only mode.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1240.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The user under which the API request was authenticated does not have the required authority to perform this operation.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/logical-partitions/d39347d9-855a-3199-9ef6-a1701b7b17b4/operations/
   list-os-messages HTTP/1.1
x-api-session: 2cckypxdonb44w6n8dvyly3vpw9hekwr2soasrhax41oetlbni
```

Figure 646. List OS Messages of a Logical Partition: Request

```
200 OK
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Mon, 28 Aug 2017 19:59:00 GMT
content-type: application/json;charset=UTF-8
content-length: 665
{
   "os-messages":[
      {
         "is-held":true,
         "is-priority":false,
         "message-id":69166,
         "message-text":"*IOS002A AB4D,NO PATHS AVAILABLE\n",
         "os-name":"S50     ",
         "prompt-text":"",
         "sequence-number":10541,
         "sound-alarm":false,
         "timestamp":1503926847000
      },
      {
         "is-held":true,
         "is-priority":false,
         "message-id":69167,
         "message-text":"*IOS002A AB11,NO PATHS AVAILABLE\n",
         "os-name":"S50     ",
         "prompt-text":"",
         "sequence-number":10542,
         "sound-alarm":false,
         "timestamp":1503926847000
      },
      {
         "is-held":true,
         "is-priority":false,
         "message-id":69169,
         "message-text":"*IOS002A AB42,NO PATHS AVAILABLE\n",
         "os-name":"S50     ",
         "prompt-text":"",
         "sequence-number":10543,
         "sound-alarm":false,
         "timestamp":1503926847000
      }
   ]
}
```

Figure 647. List OS Messages of a Logical Partition: Response

Delete Logical Partition OS Message

The `Delete Logical Partition OS Message` operation deletes a single logical partition OS message. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/delete-os-message
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target logical partition.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`sequence-number`	Long	Required	The **sequence-number** property of the OS message on the logical partition to delete.

Description

This operation deletes a specific logical partition OS message. The OS message to be deleted is uniquely identified by the combination of the *{logical-partition-id}* variable in the URI and the **sequence-number** in the request body.

The URI path must designate an existing logical partition and the API user must have access permission to it; otherwise status code 404 (Not Found) is returned.

The request body must designate an existing OS message; otherwise, status code 404 (Not Found) is returned. In addition, for the web services interface the API user must have Action/Task permission to the Operating System Messages task; otherwise, status code 403 (Forbidden) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Operating System Messages** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned with no response body provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The API user does not have action/task permission for the **Operating System Messages** task.
404 (Not Found)	1	The object ID in the URI *{logical-partition-id}* does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
	336	The **sequence-number** in the request body does not designate an existing OS message on the logical partition.
409 (Conflict)	332	The messages interface is not available.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/4767c72e-b00d-3a1c-ac89-87f821404a0b/operations/delete-os-message
   HTTP/1.1
x-api-session: 6d1q521ruuym4dhqubv3m77678qbjx7c68wpdpv5v75ut8n1iq
content-type: application/json
content-length: 22
{
   "sequence-number":0
}
```

Figure 648. Delete Logical Partition OS Message: Request

```
204 No Content
server: Hardware management console API web server / 2.0
cache-control: no-cache
date: Thu, 15 Nov 2018 17:21:00 GMT

<No response body>
```

Figure 649. Delete Logical Partition OS Message: Response

SCSI Load

The `SCSI Load` operation prepares the logical partition for loading an operating system and loads the operating system from the designated SCSI device. This operation is supported using the BCPii interface.

HTTP method and URI

`POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/scsi-load`

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`load-address`	String (1-5)	Required	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, $, #) are also allowed when the **se-version** property of the associated CPC is **"2.14.0"** or later.

Field name	Type	Rqd/Opt	Description
`secure-boot`	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. The default value is **false**. This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
`clear-indicator`	Boolean	Optional	Whether memory should be cleared before performing the `Load` (true) or not cleared (false). The default value is true. **Note:** This field is allowed only when the **se-version** property of the associated CPC is **"2.14.1"** or later.
`world-wide-port-name`	String (1-16)	Required	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal.
`logical-unit-number`	String (1-16)	Required	The hexadecimal logical unit number (LUN) to be used for the `SCSI Load`.
`disk-partition-id`	Integer (0-30)	Optional	The **disk-partition-id** (also called the boot program selector) to be used for the `SCSI Load`. The default value is 0.
`operating-system-specific-load-parameters`	String (0-256)	Optional	The operating system specific load parameters to be used for the `SCSI Load`. The default value is an empty string.
`boot-record-logical-block-address`	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the `SCSI Load`. The default value is hex zeros.
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). The default is false.
`os-ipl-token`	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-F), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1248. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1248.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1246.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Usage note

Beginning with API version 4.10 this operation will not be enhanced to support new load functionality. Instead, API users are encouraged to use “Load” on page 1218, which includes the functionality of this operation as well as additional capabilities. [Updated by feature **secure-boot-with-certificates**]

SCSI Dump

The `SCSI Dump` operation loads a standalone dump program from a designated SCSI device. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/scsi-dump
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`load-address`	String (1-5)	Required	The hexadecimal address of an I/O device that provides access to the control program to be loaded. The input value is right justified and padded with zeros to 5 characters. Valid values are in the range "00000" to "nFFFF" where "n" is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a `Load` operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, $, #) are also allowed when the **se-version** property of the associated CPC is **"2.14.0"** or later.

Field name	Type	Rqd/Opt	Description
`secure-boot`	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. The default value is **false**. This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
`world-wide-port-name`	String (1-16)	Required	The worldwide port name (WWPN) of the target SCSI device to be used for this operation, in hexadecimal.
`logical-unit-number`	String (1-16)	Required	The hexadecimal logical unit number (LUN) to be used for the `SCSI Dump`.
`disk-partition-id`	Integer (0-30)	Optional	The disk-partition-id (also called the boot program selector) to be used for the `SCSI Dump`. The default value is 0.
`operating-system-specific-load-parameters`	String (0-256)	Optional	The operating system specific load parameters to be used for the `SCSI Dump`. The default value is an empty string.
`boot-record-logical-block-address`	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the `SCSI Dump`. The default value is hex zeros.
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating"** status (true) or not (false). Default: false
`os-ipl-token`	String (1-16)	Optional	Applicable only to z/OS, this parameter requests that this operation only be performed if the provided value matches the current value of the **os-ipl-token** property. This ensures that this operation is targeting the same IPL instance as when the **os-ipl-token** property was retrieved. IBM recommends that this parameter only be provided by callers that fully understand how the **os-ipl-token** parameter is managed by z/OS. The value is a string of hexadecimal characters (0-9, A-Z), left justified.

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as

indicated in "Job status and reason codes" on page 1251. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1251.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1249.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Usage note

Beginning with API version 4.10 this operation will not be enhanced to support new load functionality. Instead, API users are encouraged to use "Load" on page 1218, which includes the functionality of this operation as well as additional capabilities. [Updated by feature **secure-boot-with-certificates**]

NVMe Load

The `NVMe Load` operation prepares the logical partition for loading an operating system and loads the operating system from the designated Non-volatile Memory Express NVMe device. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/nvme-load
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`load-address`	String (1-4)	Required	The hexadecimal address (i.e. the function ID, or FID) of an I/O device that provides access to the control program to be loaded. If the input value is less than 4 in length, it will be right justified and padded with zeros to 4 characters. The final value must be in the range "0000" to "FFFF".
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of an `NVMe Load` operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, @, $, #, blank, and period. Default: an empty string
`secure-boot`	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The operation will fail if the signatures do not match. This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. Default: false

Field name	Type	Rqd/Opt	Description
`clear-indicator`	Boolean	Optional	Whether memory should be cleared before performing the `NVMe Load` (true) or not cleared (false). Default: true
`disk-partition-id`	Integer (0-30)	Optional	The **disk-partition-id** (also called the boot program selector) to be used for the `NVMe Load`. Default: 0
`operating-system-specific-load-parameters`	String (0-256)	Optional	The operating system specific load parameters to be used for the `NVMe Load`. Default: an empty string
`boot-record-logical-block-address`	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the `NVMe Load`. Default: hex string "0"
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating" status** (true) or not (false). Default: false

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1253. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1253.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1252.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The API user does not have the required action/task permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
	4	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object whose associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Usage note

Beginning with API version 4.10 this operation will not be enhanced to support new load functionality. Instead, API users are encouraged to use "Load" on page 1218, which includes the functionality of this operation as well as additional capabilities. [Updated by feature **secure-boot-with-certificates**]

NVMe Dump

The `NVMe Dump` operation loads a standalone dump program from a designated Non-volatile Memory Express (NVMe) device. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/nvme-dump
```

In this request, the URI variable *{logical-partition-id}* is the object ID of the target Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`load-address`	String (1-4)	Required	The hexadecimal address (i.e. the function ID, or FID) of an I/O device that provides access to the control program to be loaded. If the input value is less than 4 in length, it will be right justified and padded with zeros to 4 characters. The final value must be in the range "0000" to "FFFF".
`load-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of an `NVMe Dump` operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. Valid characters are 0-9, A-Z, @, $, #, blank, and period. Default: an empty string
`secure-boot`	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The operation will fail if the signatures do not match. This field is allowed only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. Default: false
`disk-partition-id`	Integer (0-30)	Optional	The **disk-partition-id** (also called the boot program selector) to be used for the `NVMe Dump`. Default: 0
`operating-system-specific-load-parameters`	String (0-256)	Optional	The operating system specific load parameters to be used for the `NVMe Dump`. Default: an empty string
`boot-record-logical-block-address`	String (1-16)	Optional	The hexadecimal boot record logical block address to be used for the `NVMe Dump`. Default: hex string "0"
`force`	Boolean	Optional	Whether this operation is permitted when the logical partition is in **"operating" status** (true) or not (false). Default: false

Response body contents

Once the operation is accepted, the response body contains a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String/ URI	URI that may be queried to retrieve status updates.

Asynchronous result description

Once the operation has completed, a job-completion notification is sent and results are available for the asynchronous portion of this operation. These results are retrieved using the `Query Job Status` operation directed at the job URI provided in the response body.

The result document returned by the `Query Job Status` operation is specified in the description for the `Query Job Status` operation. When the status of the job is **"complete"**, the results include a job completion status code and reason code (fields **job-status-code** and **job-reason-code**) which are set as indicated in "Job status and reason codes" on page 1256. The **job-results** field is null when this operation is successful. When it is partially successful or not successful, the **job-results** field contains an object with the following field:

Field name	Type	Description
message	String	The message text describing the detailed error that occurred when the operation was partially successful or not successful.

Description

When the operation is initiated, a 202 (Accepted) status code is returned. The response body includes a URI that may be queried to retrieve the status of the operation. See "Query Job Status" on page 151 for information on how to query job status. When the operation has completed, an asynchronous result message is sent. See "Job status and reason codes" on page 1256.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the Logical Partition object designated by *{logical-partition-id}*
  - Action/task permission for the **Load** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1255.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object.
	1	The API user does not have the required action/task permission for this operation.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
	4	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object whose associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.
500 (Server Error)	280	An IO exception occurred during the scheduling of the asynchronous request.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

HTTP error status code	Reason code	Description
204 (No Content)	N/A	Operation completed successfully.
500 (Server Error)	263	Operation failed.

Usage note

Beginning with API version 4.10 this operation will not be enhanced to support new load functionality. Instead, API users are encouraged to use "Load" on page 1218, which includes the functionality of this operation as well as additional capabilities. [Updated by feature **secure-boot-with-certificates**].

Assign Certificate to Logical Partition

The `Assign Certificate to Logical Partition` operation assigns a certificate of **type "secure-boot"** to a logical partition. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/assign-certificate
```

In this request, the URI variable *{logical-partition-id}* is the Object ID of the Logical Partition object.

Request body contents

The request body is a JSON object with the following field:

Table 526.

Field name	Type	Rqd/ Opt	Description
`certificate-uri`	String/ URI	Required	The URI of the certificate to be assigned.

Description

This operation assigns a secure boot certificate to a logical partition.

If the logical partition specified cannot be assigned certificates, because it is a system defined logical partition, a 400 (Bad Request) status code is returned. A 404 (Not Found) status code is returned if the request does not designate an existing logical partition, CPC, or Certificate, or if the API user does not have object-access permission to the object. If the API user doesn't have action/task permission to the **Assign Secure Boot Certificates** task, 403 (Forbidden) status code is returned. If the Certificate object is currently assigned to the logical partition, if assigning the Certificate would exceed the Certificate limit of 20 per profile, or if attempting to assign to an unmanaged CPC, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the target object.
  - Object-access permission to the Certificate object whose **object-id** is *{certificate-id}*.
  - Action/task permission for the **Assign Secure Boot Certificates** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	372	The operation could not be performed because this certificate would exceed limit of 20 certificates per partition.
	373	The operation cannot be performed because the certificate has already been assigned to this logical partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/43ecac48-f190-38b7-b61d-a520efd3296e/operations/assign-certificate
HTTP/1.1
x-api-session: 3ypnm33mkdg2xso9l52mk714g3gr55zi661z7wzhzurrpzh5t3
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/ab07f6ca-402f-11ed-ab57-fa163e6f7e7e"
}
```

Figure 650. Assign Certificate to Logical Partition: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 11 Oct 2022 15:22:02 GMT

<No response body>
```

Figure 651. Assign Certificate to Logical Partition: Response

Unassign Certificate from Logical Partition

The `Unassign Certificate from Logical Partition` operation unassigns a certificate of **type "secure-boot"** from an logical partition. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/unassign-certificate
```

In this request, thye URI variable *{logical-partition-id}* is the object ID of the Logical Partition object.

Request body contents

The request body is a JSON object with the following field:

Field name	Type	Description
`certificate-uri`	String/ URI	The URI of the certificate to be unassigned.

Description

This operation unassigns a secure boot certificate from a logical partition.

If certificates cannot be unassigned from the logical partition specified because it is a system defined logical partition, a 400 (Bad Request) status code is returned. A 404 (Not Found) status code is returned if the request does not designate an existing logical partition, CPC, or Certificate, or if the API user does not have object-access permission to the object. If the API user does not have action/task permission to the **Assign Secure Boot Certificates** task, 403 (Forbidden) status code is returned. If the Certificate object is not currently assigned to the logical partition, or if attempting to unassign from an unmanaged CPC, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the target object.
  - Object-access permission to the certificate object whose **object-id** is *{certificate-id}*.
  - Action/task permission for the **Assign Secure Boot Certificates** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	370	The operation cannot be performed because the certificate is not assigned to this logical partition.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/43ecac48-f190-38b7-b61d-a520efd3296e/operations/unassign-
certificate HTTP/1.1
x-api-session: 151s9hi1uyfntd6woaefccrv1163mvxdeef8ztc0v95mykd1uw
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/ab07f6ca-402f-11ed-ab57-fa163e6f7e7e"
}
```

Figure 652. Unassign Certificate from Logical Partition: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 11 Oct 2022 15:21:19 GMT

<No response body>
```

Figure 653. Unassign Certificate from Logical Partition: Response

Report a Logical Partition Problem

The `Report a Logical Partition Problem` reports and requests service for a problem on a Logical Partition object designated by *{logical-partition-id}*. This operation is supported using the BCPii interface. [Added by feature **report-a-problem**]

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/report-problem
```

In this request, the URI variable *{logical-partition-id}* is the Object ID of the Logical Partition object.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Name	Type	Req/Opt	Description
`customer-name`	String (0-50)	Optional	Name of the customer. May not contain any double-byte characters or ";". Default: "Unknown"
`customer-phone-number`	String (0-20)	Optional	Phone number of customer. May not contain any double-byte characters or ";". Default: "Unknown"
`problem-description`	String (1-510)	Required	Description of the problem. May not contain any double-byte characters or ";".

Name	Type	Req/Opt	Description
`problem-type`	String Enum	Required	Identifies the type of problem. One of: • **"power"** - Report a problem with the power subsystem. • **"cpc"** - Report a problem with hardware in the processor subsystem. • **"lan"** - Report a problem with the local area network (LAN). • **"software"** - Report a problem with an operating system or other software. • **"io"** -Report a problem with hardware in the input/output (I/O) configuration. • **"health"** - Report the state of the system before applying a maintenance action. • **"other"** - Report a problem that is not adequately described by any other problem type. • **"test"** - Test whether problems can be reported for the selected system.

Description

The `Report a Logical Partition Problem` operation reports a problem for a Logical Partition object and requests service to repair it.

Problems are reported to the support system for the provided system. Reporting a problem sends the information provided in the request and the machine information that identifies the system to the service provider.

Automatic service call reporting must be enabled on the SE associated with the Logical Partition object via the **Remote Service** task to use this operation. If the SE associated with the Logical Partition object does not have automatic service call reporting enabled, a 409 (Conflict) status code is returned.

Upon successful problem creation, a 204 (No Content) status code is returned. If the API user does not have action/task permission to the **Report a Problem** task, a 403 (Forbidden) status code is returned. If the SE associated with the Logical Partition object is unreachable, a 503 (Service Unavailable) status code is returned. The URI path must designate an existing Logical Partition and the API user must have object-access permission to it; otherwise, status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*.
- Action/task permission for the **Report A Problem** task.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the Logical Partition object
	1	The API user does not have the required action/task permissions.
404 (Not Found)	1	The object ID in the URI (*{logical-partition-id}*) does not designate an existing Logical Partition object, or the API user does not have object-access permission to the object.
409 (Conflict)	600	The operation cannot be performed because the SE does not have automatic service call reporting enabled.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/4dd547b3-0038-3966-921c-0737ae39b1e2/operations/report-problem
HTTP/1.1
x-api-session: 1juit65mrqbaa7ld18276tbu89bzsuoujj7t263uaclfcblvgi
Content-Type: application/json
Content-Length: 142
{
  "customer-name":"Tester",
  "customer-phone-number":"888-888-8888",
  "problem-description":"This is a test IO problem",
  "problem-type":"io"
}
```

Figure 654. Report a Logical Partition Problem: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 06 Feb 2023 23:28:20 GMT
<No response body>
```

Figure 655. Report a Logical Partition Problem: Response

Get Logical Partition Historical Sustainability Data

Use the `Get Logical Partition Historical Sustainability Data` operation to retrieve logical partition data on a specific time range. This operation is supported using the BCPii interface. [Added by feature **environmental-metrics**]

HTTP method and URI

```
POST /api/logical-partitions/{logical-partition-id}/operations/get-historical-sustainability-data
```

In this request, the URI variable *{logical-partition-id}* is the Object ID of the Logical Partition object.

Request body contents

The request body is a JSON object with the following fields:

Field Name	Type	Rqd/Opt	Description
`range`	String Enum	Optional	Time range for the historical data points. This is the amount of time to be covered by all data points. The possible values are as follows: • **"last-day"** - Last 24 hours. • **"last-week"** - Last 7 days. • **"last-month"** - Last 30 days. • **"last-three-months"** - Last 90 days. • **"last-six-months"** - Last 180 days. • **"last-year"** - Last 365 days. • **"custom"** - From **custom-range-start** to **custom-range-end**. If not specified, the default value is **"last-week"**.
`custom-range-start`	Timestamp	Required if **range** is **"custom"**	Start time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than or equal to 0.
`custom-range-end`	Timestamp	Required if **range** is **"custom"**	End time in custom range for the historical data points. This is specified as the number of milliseconds since the epoch and must be greater than custom-range-start.
`resolution`	String Enum	Optional	Resolution of requested data points. This is the time interval in between data points. For systems where the "environmental-metrics" feature is not available, the minimum resolution is **"one-hour"**. The possible values are as follows: • **"fifteen-minutes"**- 15 minutes. • **"one-hour"**- 60 minutes. • **"one-day"**- 24 hours. • **"one-week"** - 7 days. • **"one-month"** - 30 days. If not specified, the default value is **"one-hour"**.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
`wattage`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing the estimated power consumed by the partition, in Watts, at a specific point in time.
`processor-utilization`	Array of integer-data-point objects	Each element of this array is an integer-data-point object representing partition processor utilization in percentage, at a specific point in time.

Description

This operation returns an array of available historical data points in a logical partition designated by *{logical-partition-id}*.

If the **range** field in the request body content is not **"custom"**, **custom-range-start** and **custom-range-end** are ignored and can be omitted from the request. Otherwise, those fields need to be set or HTTP status code 400 (Bad Request) is returned. Additionally, both need to be greater than zero or HTTP status code 400 (Bad Request) will be returned. Finally, **custom-range-end** must be greater than **custom-range-start** or else HTTP status code 400 (Bad Request) is returned. Should the custom range be greater than the existing range of measured data, the operation will complete successfully and return an array with the existing data points.

On successful execution, HTTP status code 200 (OK) is returned with the response body containing properties defined in “Response body contents” on page 1264. Should the "environmental-metrics" feature not be available on the HMC, HTTP status code 404.1 (Not Found) is returned. If the same feature is not available on the CPC, HTTP status code 404.4 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Logical Partition object designated by *{logical-partition-id}*.
- Action/task permission to the **Environmental Dashboard** task.

For the BCPii interface the source partition must have receive BCPii security controls permissions for the Logical Partition object designated by *{logical-partition-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in “Response body contents” on page 1264.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and the associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The data type of a field in the request body is not as expected or its value is not in the permitted range.
	15	The request body contains a field whose presence or value is inconsistent with the presence or value of another field in the request body.
404 (Not Found)	1	The request URI does not designate a resource of an expected type or designates a resource for which the user does not have permission.
	4	The object designated by the request URI does not support the requested operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/logical-partitions/194200f8-9a0d-3752-bc1a-68f3f5a7b518/operations/get-historical-
sustainability-data HTTP/1.1
x-api-session: pewb4388kwumqgpywy55hwdxl1l0fgxfjeb9lb22oh9c7x0il
Content-Type: application/json
Content-Length: 46
{
   "range":"last-day",
   "resolution":"one-day"
}
```

Figure 656. Get Logical Partition Historical Sustainability Data: Request

```
200 OK
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 15 May 2023 16:28:05 GMT
Content-Type: application/json
Content-Length: 115
{
   "processor-utilization":[
      {
         "data":100,
         "timestamp":1684109336255
      }
   ],
   "wattage":[
      {
         "data":254,
         "timestamp":1684109336255
      }
   ]
}
```

Figure 657. Get Logical Partition Historical Sustainability Data: Response

Inventory service data

Information about logical partitions can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Logical Partition objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"logical-partition"** are to be included. An entry for a particular logical partition is included only if the API user has access permission to that object as described in the `Get Logical Partition Properties` operation.

For each Logical Partition object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Logical Partition Properties" on page 1200. That is, the data provided is the same as would be provided if a `Get Logical Partition Properties` operation were requested targeting this object.

Certificate object

The Certificate managed object represents an X.509 certificate with a wide area of applicable uses such as code and document signing, authenticating digital entities of devices, people, data, and applications, TLS/SSL and web browser security, authenticating digital signatures, and much more. There is no Create Certificate operation. Instead, certificates of **type "secure-boot"** are created via the `Import CPC Certificate` operation. [Added by feature **secure-boot-with-certificates**]

Data model

This object includes the properties defined in the "Base managed object properties schema" on page 100 but does not provide the operational-status-related properties defined in that schema because it does not maintain the concept of an operational status

Table 527. Certificate object: base managed object properties specializations

Name	Qualifier	Type	Description of specialization
object-uri	—	String/ URI	The canonical URI path of the Certificate object, of the form `/api/certificates/{certificate-id}` where *{certificate-id}* is the value of the **object-id** property of the Certificate object.
parent	—	String/ URI	The parent of a certificate, whose **type** is **"secure boot"**, is conceptually the CPC that it is imported to, and so the parent value is the canonical URI path for the CPC.
name	(w)(pc)	String (1-64)	The display name specified for the certificate. The character requirements on this property are the same as those of the name property described in the "Base managed object properties schema" on page 100. Names must be unique among all certificates with the same parent and type values.
description	(w)(pc)	String (0-1024)	The description of the certificate. Default: An empty string
class	—	String (11)	The class of a Certificate object is **"certificate"**.

Class specific additional properties

In addition to the properties defined through included schemas, this object includes the following additional class-specific properties:

Table 528. Certificate object: class specific additional properties			
Name	**Qualifier**	**Type**	**Description**
parent-name	(a)	String (1-64)	The name of the object identified by the **parent** property.
type	—	String Enum	The type of certificate with regards to its intended usage, associated task, or associated action. Valid values: • **"secure-boot"** - the certificate is used for the **Secure Boot Certificate Management** task.
sha-256-fingerprint	—	String	The SHA-256 fingerprint of the certificate.
assigned	(pc)	Boolean	For certificates of **type "secure-boot"**, identifies if the certificate is assigned to one or more logical partitions, partitions, or image activation profiles.

Delete Certificate

The `Delete Certificate` operation deletes the identified certificate. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`DELETE /api/certificates/{certificate-id}`

In this request, the URI variable *{certificate-id}* is the object ID of the Certificate object to be deleted.

Description

This operation deletes the specified certificate. Upon success, an Inventory Change notification is emitted asynchronously to this operation.

If this operation changes the value of any property for which property-change notifications are due, those notifications are issued asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing Certificate object, or if the API user does not have object-access permission to the object. If the API user doesn't have action/task permission to the Import Secure Boot Certificates task, 403 (Forbidden) status code is returned. If the Certificate object is currently assigned, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the certificate object whose **object-id** is *{certificate-id}*.
- Action/task permission to the **Import Secure Boot Certificates** task.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by **target-name** property that accompanies the request.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source CPC object does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{certificate-id}*) does not designate an existing Certificate object, or the API user does not have object-access permission to the object.
409 (Conflict)	373	The operation cannot be performed because the certificate is currently assigned.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/certificates/dab30826-48d4-11ed-87c1-fa163e6f7e7e HTTP/1.1
x-api-session: 3x0ewon9h1e6isqpylay1qphjgh4t6f3xeohmfqtcpwymgfkzy
```

Figure 658. Delete Certificate: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 20:03:41 GMT
Content-Type: application/json

<No response body>
```

Figure 659. Delete Certificate: Response

Get Certificate Properties

The `Get Certificate Properties` operation retrieves the properties of a single Certificate object. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`GET /api/certificates/{certificate-id}`

In this request, the URI variable *{certificate-id}* is the value of the **object-id** of the Certificate object.

Query parameters

Name	Type	Rqd/Opt	Description
properties	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the Certificate object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
object-uri	String/ URI	The canonical URI path of the Certificate object, of the form /api/certificates/*{certificate-id}* where *{certificate-id}* is the value of the **object-id** property of the Certificate object.
parent	String/ URI	The parent of a certificate, whose **type** is **"secure boot"**, is conceptually the CPC that it is imported to, and so the parent value is the canonical URI path for the CPC.
parent-name	String (1 - 64)	The name of the object identified by the **parent** property.
name	String (1 - 64)	The display name specified for the certificate. The character requirements on this property are the same as those of the **name** property described in the "Base managed object properties schema" on page 100. Names must be unique among all Certificates with the same parent and type values.
description	String (0 - 1024)	The description of the certificate.
class	String (11)	The **class** of a Certificate object is **"certificate"**.
type	String Enum	The type of certificate with regards to its intended usage, associated task, or associated action. Valid values: • **"secure-boot"** - The certificate is used for the **Secure Boot Certificate Management** task.
sha-256-fingerprint	String	The SHA-256 fingerprint of the certificate.
assigned	Boolean	For certificates of **type secure-boot**, identifies if the certificate is assigned to one or more logical partitions, partitions, or image activation profiles.

Description

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Certificate object.

A 404 (Not Found) status code is returned if the request URI does not designate an existing Certificate object, or if the API user does not have object-access permission to the object. A 400 (Bad Request)

status code is returned if an unrecognized, unsupported, malformed or invalid query parameter was specified.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Certificate object whose **object-id** is *{certificate-id}*.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by **target-name** property that accompanies the request.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1269

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	1	The request included an unrecognized or unsupported query parameter.
	14	Query parameters on the request are malformed or specify a value that is invalid for this operation.
403 (Forbidden)	0	The request used the BCPii interface and the source CPC object does not have receive BCPii security controls permission.
404 (Not Found)	1	The object ID in the URI (*{certificate-id}*) does not designate an existing Certificate object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/certificates/471a0dfe-4031-11ed-a545-fa163e6f7e7e HTTP/1.1
x-api-session: d4fwcg8f4jdwfkyzjfuhh5igbmk0efz1glxrszia7b0a0r3j5
```

Figure 660. Get Certificate Properties: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 19:23:35 GMT
Content-Type: application/json
Content-Length: 482
{
   "assigned":true,
   "class":"certificate",
   "description":"Certificate for secure boot with new z/OS",
   "name":"zOS validated boot certificate",
   "object-id":"471a0dfe-4031-11ed-a545-fa163e6f7e7e",
   "object-uri":"/api/certificates/471a0dfe-4031-11ed-a545-fa163e6f7e7e",
   "parent":"/api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4",
   "parent-name":"HJVS2EKN",
   "sha-256-fingerprint":"0C ED 78 C4 80 2B 2B 9A 3D 19 0F 75 8A 79 F0 05 87 EF 22 94 69 D6 80
      A0 C6 3B 2F EE D3 12 0D 83",
   "type":"secure-boot"
}
```

Figure 661. Get Certificate Properties: Response

Get Encoded Certificate

The `Get Encoded Certificate` operation retrieves the Base64 encoded string and the file format of the certificate. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`GET /api/certificates/{certificate-id}/operations/get-encoded`

In this request, the URI variable *{certificate-id}* is the value of the **object-id** property of the Certificate object.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
`certificate`	String	The Base64 encoded string of the certificate.
`format`	String Enum	The format of the certificate. One of: • **"der"** - Distinguished Encoding Rules • **"pem"** - Privacy Enhanced Mail

Description

This operation gets the Base64 encoded string and the file format of the certificate specified by *{certificate-id}*. On successful execution, the Base64 encoded string and the file format of the certificate is provided in the response body, and HTTP status code 200 (OK) is returned

A 404 (Not Found) status code is returned if the request URI does not designate an existing Certificate object, or if the API user does not have object-access permission to the object.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Certificate object whose **object-id** is *{certificate-id}*.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by **target-name** property that accompanies the request.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1271

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source CPC object does not have receive BCPii security controls permission.
404 (Not Found)	1	The object ID in the URI (*{certificate-id}*) does not designate an existing Certificate object, or the API user does not have object-access permission to the object.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/certificates/471a0dfe-4031-11ed-a545-fa163e6f7e7e/operations/get-encoded HTTP/1.1
x-api-session: 3q5sdnofqf5g7p1zikwpyqmi5keu74zry70f9qrnvtsporum1j
```

Figure 662. Get Encoded Certificate: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 19:40:57 GMT
Content-Type: application/json
Content-Length: 934
{

"certificate":"MIICmzCCAf6gAwIBAgIBATAKBggqhkjOPQQDBDBKMQswCQYDVQQGEwJVUzERMA8GA1UEChMISUJNIENvc
nAxKDAmBgNVBAsTH0hNQyB0byBQUi9TTSBDb2RlIFNpZ25pbmcgQ0EgVkIwHhcNMjIwMTA0MDUwMDAwWhcNMjcwMTAxMDQ1O
TU5WjBHMQswCQYDVQQGEwJVUzERMA8GA1UEChMISUJNIENvcnAxJTAjBgNVBAMTHEhNQyB0byBQUi9TTSBDb2RlIFNpZ25pb
mcgVkIwgZswEAYHKoZIzj0CAQYFK4EEACMDgYYABAD7NLnkNWNyPfNdklymhyoFUDcjRI/
cKESkJbZI6qdaQDL4Hvx3tiuOLZAykaIbvKqVC2DpTqPfrQ2wBNCGbGhw6QDFsleVhKQ3fBfvoNjNvMvdhDd6+SYbhgYOdlb
BDQeHCrWHvTLHEYVhDUYfjlEv69JDzA3IBvFEpO7L++MBCfRFdaOBlDCBkTA/
BglghkgBhvhCAQ0EMhYwR2VuZXJhdGVkIGJ5IHRoZSBTZWN1cml0eSBTZXJ2ZXIgZm9yIHovT1MgKFJBQ0YpMA4GA1UdDwEB
/
wQEAwIHgDAdBgNVHQ4EFgQUIcyV0IoS+f5aoBWYQw72oI1Y394wHwYDVR0jBBgwFoAUwXMRSBSyS9mOu8+Tirb2I84pphYwC
gYIKoZIzj0EAwQDgYoAMIGGAkEhjs1yBG1CRUhz94XiyCVctURjC5IUBVKGX4spm9gMnhUxSVurl7XonxNeOdlSiRG5t8wHk
yDm9/
ET55SAIHSnLAJBFbOUeW45+6+qEDeLT02fv19wR+l+DzFOKPRoarEpnVE0HFWn3wBgZK8tjlvg7rTuf1LGAe6QjICPFqzhAy
7Ur0A=",
   "format":"der"
}
```

Figure 663. Get Encoded Certificate: Response

Usage note

This allows the ability to "copy" a Certificate from one CPC to another. This also allows the ability to view more properties than are provided in the base Certificate Data Model.

List Certificates

The `List Certificates` operation lists the Certificate objects known to the Console. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`GET /api/certificates`

Query parameters

Name	Type	Rqd/Opt	Description
`name`	String	Optional	Filter pattern (regular expression) to limit returned object to thos that have a matching **name** property.
`type`	String Enum	Optional	Filter string to limit returned objects to those that have a matching **type** property. Value must be a valid certificate **type** property value.
`parent-name`	String	Optional	Filter pattern (regular expression) to limit returned objects to those that have a matching **parent-name** property.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties **name**, **type**, **parent-name**, **object-uri**, **parent**. This is a list of comma-separated strings where each string is a property name defined in the Certificate object's data model.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field Name	Type	Description
`certificates`	Array of certificate-info objects	Array of nested objects (described in the next table).

Each nested certificate-info object contains the following fields:

Field Name	Type	Description
`object-uri`	String	The **object-uri** property of the Certificate object.
`name`	String	The **name** property of the Certificate object.
`type`	String	The **type** property of the Certificate object.
`parent-name`	String	The name of the object identified by the **parent** property of the Certificate object.
`parent`	String	The **parent** property of the Certificate object.

Description

For the web services interface this operation lists the Certificate objects to which the API user has object-access permission. For the BCPii interface this operation lists the Certificate objects to which the source partition has receive BCPii security controls permissions. Some basic properties are provided for each certificate that is included in the response.

If the **name** query parameter is selected, the returned list is limited to those certificates that have a **name** property matching the specified filter pattern. If the **name** parameter is omitted, no such filtering is performed.

if the **type** query parameter is selected, the returned list is limited to those certificates that have a **type** property matching the specified value. If the **type** parameter is omitted, no such filtering is performed.

If the **parent-name** query parameter is selected, the returned list is limited to those certificates that have a **parent-name** property matching the specified filter pattern. If the **parent-name** parameter is omitted, no such filtering is performed.

If the **additional-properties** parameter is selected, additional properties are included in the returned list. The properties to be included is a list of comma-separated strings where each string is a property name defined in the Certificate object's data model. If the **additional-properties** parameter is omitted, no such properties will be included.

If there are no Certificates, an empty list is provided and the operation completes successfully.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the Certificate objects included in the response body.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by **target-name** property that accompanies the request.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1273

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	1	The request included an unrecognized or unsupported query parameter.
	14	Query parameters on the request are malformed or specify a value that is invalid for this operation.
403 (Forbidden)	0	The request used the BCPii interface and the source CPC object does not have receive BCPii security controls permission.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/certificates HTTP/1.1
x-api-session: 5orfeb8smkhf3xpnqaj9pjv7m1upv34gjy03uevsadyw0c80du
```

Figure 664. List Certificates: Request

```
200
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 19:20:19 GMT
Content-Type: application/json
Content-Length: 437
{
   "certificates":[
      {
         "name":"zOS validated boot certificate",
         "object-uri":"/api/certificates/471a0dfe-4031-11ed-a545-fa163e6f7e7e",
         "parent":"/api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4",
         "parent-name":"HJVS2EKN",
         "type":"secure-boot"
      },
      {
         "name":"Linux certificate",
         "object-uri":"/api/certificates/ab07f6ca-402f-11ed-ab57-fa163e6f7e7e",
         "parent":"/api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4",
         "parent-name":"HJVS2EKN",
         "type":"secure-boot"
      }
   ]
}
```

Figure 665. List Certificates: Response

Update Certificate Properties

The `Update Certificate Properties` operation modifies the properties of a single Certificate object. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

`POST /api/certificates/`*`{certificate-id}`*

In this request, the URI variable *{certificate-id}* is the value of the **object-id** property of the Certificate object.

Request body contents

The request body is expected to contain one or more field names representing writable certificate properties, along with the new values for those fields. The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Certificate to ensure that the request body contains only writable properties and the data types of those properties are as required. The request body can and should omit fields for properties whose values are not to be changed by this operation. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Certificate is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned. When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

A 404 (Not Found) status code is returned if the request URI does not designate an existing Certificate object, or if the API user does not have object-access permission to the object. If the API user doesn't have action/task permission to Certificate Details task, 403 (Forbidden) status code is returned. A 400 (Bad Request) status code is returned if the name field is not unique among all Certificates with the same parent and type values. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the parent CPC.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/task permission to the **Certificate Details** task.
- Object-access permission to the Certificate object whose **object-id** is *{certificate-id}*.

For the BCPii interface:

- The source partition must have receive BCPii security controls permissions for the CPC object designated by **target-name** property that accompanies the request.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The value of a field does not provide a unique value for the corresponding data model property as required.
403 (Forbidden)	0	The request used the BCPii interface and the source CPC object does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{certificate-id}*) does not designate an existing Certificate object, or the API user does not have object-access permission to the object.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST/api/certificates/dab30826-48d4-11ed-87c1-fa163e6f7e7e HTTP/1.1
x-api-session: 2murd8f88nv5u27uw29vbenpm5vzed27ih9fpep564dw71hasv
Content-Type: application/json
Content-Length: 81
{
   "description":"Certificate for Linux secure boot",
   "name":"Linux Certificate"
}
```

Figure 666. Update Certificate Properties: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Mon, 10 Oct 2022 19:54:46 GMT
Content-Type: application/json

<No response body>
```

Figure 667. Update Certificate Properties: Response

Inventory service data

Information about the Certificate objects managed by the HMC can be optionally included in the inventory data provided by the Inventory Service. [Added by feature **secure-boot-with-certificates**]

Inventory entries for Certificate objects are included in the response to the Inventory Service's `Get Inventory` operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of **class "secure-boot-certificate"** are to be included. The inventory **class "secure-boot-certificate"** includes Certificate objects with a **type** value of **"secure-boot"**. Information for a particular certificate is included only if the API user has object-access permission to that object.

For each certificate to be included, the inventory response includes an array entry for the Certificate object itself. This entry is a JSON object with the same contents as is specified in the Response body contents section for "Get Certificate Properties" on page 1268. That is, the data provided is the same as would be provided if a `Get Certificate Properties` operation were requested targeting this object.

Sample inventory data

The following fragment is an example of the JSON object that would be included in the `Get Inventory` response to describe a single user. This object would appear as one array entry in the response array:

```
{
   "assigned":false,
   "class":"certificate",
   "description":"",
   "name":"zOS validated boot certificate",
   "object-id":"aefc04aa-53d1-11ed-a902-fa163e43c5f5",
   "object-uri":"/api/certificates/aefc04aa-53d1-11ed-a902-fa163e43c5f5",
   "parent":"/api/cpcs/eb73e491-9544-3645-a0d9-2d1389a02066",
   "parent-name":"FL65E2GX",
   "sha-256-fingerprint":"10 2B 99 1E D8 C6 8F B0 1E 8B AA 0B 3E AF FB 9C
      AA 35 56 8B 6B 52 CC 87 19 46 D6 29 39 5F FE 5B",
   "type":"secure-boot"
}
```

Figure 668. User object: Sample inventory data

Reset activation profile

A Reset activation profile is used by a CPC Activate operation to control the activation of a CPC and, if properly configured with one or more image activation profiles, a set of Logical Partition(s).

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This element includes the following properties.

Table 529. Reset activation profile: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Reset Activation Profile object, of the form `/api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}` where *{reset-activation-profile-name}* is the value of the name property (Reset Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object.
class	—	String	The class of a Reset Activation Profile object is **"reset-activation-profile"**.
name	(pc)	String (1-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by *{cpc-id}*.
description	(w)	String (1-50)	The reset profile description
iocds-name	(w)(pc)	String (0-2)	The Input/Output Configuration Data Set name, in hexadecimal. An empty string indicates that the currently active IOCDS will be used. The active IOCDS is the one from the most recent power-on-reset of the CPC or, if using dynamic I/O configuration, the one last activated.
processor-running-time-type	(w)	String Enum	Defines whether the processor running time is determined dynamically or set manually for the CPC (see processor-running-time in this table). One of: • **"system-determined"** • **"user-determined"**
processor-running-time	(w)	Integer (0-100)	Amount of continuous time, in milliseconds, for logical processors to perform jobs on shared processors for the CPC, if **processor-running-time-type** is set to **"user-determined"**. If **processor-running-time-type** is **"system-determined"**, this property's value will always be returned as 0.

Table 529. Reset activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
end-timeslice-on-wait	(w) or — if **se-version** is **"2.14.0"** or later.	Boolean	If true and if **processor-running-time-type** is set to **"user-determined"**, CPC Logical Partitions lose their share of running time when they enter a wait state. If **processor-running-time-type** is **"system-determined"**, this property's value will always be returned as false. **Note:** When the **se-version** property of the associated CPC is **"2.14.0"** or later, this property is not permitted on an `Update Reset Activation Profile Properties` operation, and its value is always **false**.
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

List Reset Activation Profiles

The `List Reset Activation Profiles` operation lists the Reset Activation Profiles associated with a particular CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/reset-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
`name`	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`reset-activation-profiles`	Array of reset-actprof-info objects	Array of nested objects (described in the following table).

Each reset-actprof-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path of the Reset Activation Profile object.

Field name	Type	Description
`name`	String	The name of the Reset Activation Profile.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Reset Activation Profiles associated with a particular CPC.

If the **name** query parameter is specified, the returned list is limited to those Reset Activation Profiles that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1279.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1279.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 669. List Reset Activation Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:16 GMT
content-type: application/json;charset=UTF-8
content-length: 372
{
   "reset-activation-profiles": [
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-
profiles/
           DEFAULT",
         "name": "DEFAULT"
      },
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-
profiles/
           POWER_ON_RESET",
         "name": "POWER_ON_RESET"
      }
   ]
}
```

Figure 670. List Reset Activation Profiles: Response

Get Reset Activation Profile Properties

The `Get Reset Activation Profile Properties` operation retrieves the properties of a single Reset Activation Profile designated by *{reset-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

GET /api/cpcs/*{cpc-id}*/reset-activation-profiles/*{reset-activation-profile-name}*

URI variables:

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{reset-activation-profile-name}	Reset Activation Profile name

Query parameters:

Name	Type	Rqd/Opt	Description
`cached-acceptable`	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**. The default is **false**.

Name	Type	Rqd/Opt	Description
properties	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the object's data model.

Response body contents

On successful completion, the response body provides the current values of the properties for the Reset Activation Profile as defined in the "Data model" on page 1278.

Description

The URI path must designate an existing Reset Activation Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some of this object's property values are periodically fetched from the Support Element and cached for quick access by the APIs. Due to the nature of this caching support, the cached value of a property may differ from the actual value at any point in time. While the cache is kept reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Reset Activation Profile object.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1282.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{reset-activation-profile-name}*) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles/DEFAULT HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 671. Get Reset Activation Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 384
{
   "class": "reset-activation-profile",
   "description": "This is the default Reset profile.",
   "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/reset-activation-profiles/
     DEFAULT",
   "end-timeslice-on-wait": false,
   "iocds-name": "a0",
   "name": "DEFAULT",
   "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
   "processor-running-time": 0,
   "processor-running-time-type": "system-determined"
}
```

Figure 672. Get Reset Activation Profile Properties: Response

Update Reset Activation Profile Properties

The `Update Reset Activation Profile Properties` operation updates one or more writable properties of the Reset Activation Profile designated by *{reset-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}
```

URI variables:

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{reset-activation-profile-name}	Reset Activation Profile name

Request body contents

The request body is expected to contain one or more field names representing writable Reset Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Reset Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Reset Activation Profile is updated with the value provided by the input field, and status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Customize/Delete Activation Profiles** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	19	The request body contains a field whose corresponding data model property is not writable on this HMC and/or SE version.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{reset-activation-profile-name}*) does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Create Reset Activation Profile

The `Create Reset Activation Profile` operation creates a new Reset activation profile. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/reset-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

Name	Type	Rqd/ Opt	Description
`profile-name`	String (1-16)	Required	The Reset activation profile name, which uniquely identifies this profile within the set of Reset activation profiles for the CPC object designated by *{cpc-id}*.
`copy-name`	String (1-16)	Optional	The name of an existing Reset activation profile on the CPC object designated by *{cpc-id}*. If this field is provided, the name must be a valid name of an existing profile, which will then be loaded as the initial values for the Create operation. If this field is not sent in the request, the new reset profile will use the fields from the DEFAULT reset profile as the initial values for the created profile.

Name	Type	Rqd/ Opt	Description
`description`	String (0-50)	Optional	The reset profile description of the profile to be created.
`iocds-name`	String (0-2)	Optional	The Input/Output Configuration Data Set name, in hexadecimal. An empty string indicates that the currently active IOCDS will be used. The active IOCDS is the one from the most recent power-on-reset of the CPC or, if using dynamic I/O configuration, the one last activated (See the CPC Object's Data Model property **last-used-iocds**). This value is not case-sensitive.
`load-delay`	Integer (0-6000)	Optional	The delay (in seconds) before performing a Load.
`allow-expansion`	Boolean	Optional	Allow dynamic IOCDS expansion flag.
`io-priority-queuing`	Boolean	Optional	Enable global I/O Priority Queuing flag.
`global-interface-reset`	Boolean	Optional	Enable global interface reset flag.
`processor-running-time-type`	String Enum	Optional	Defines whether the processor running time is determined dynamically or set manually for the CPC (see **processor-running-time** in this table). One of: • **"system-determined"** – The processor running time is determined dynamically by the system. • **"user-determined"** – The processor running time is set manually by the user.
`processor-running-time`	Integer (0-100)	Optional	Amount of continuous time, in milliseconds, for logical processors to perform jobs on shared processors for the CPC, if **processor-running-time-type** is set to **"user-determined"**. If **processor-running-time-type** is **"system-determined"**, this value must be zero.
`display-fenced-book-page`	Boolean	Optional	Display fenced book page flag.
`how-fenced-determined`	String Enum	Optional	How are fenced book values determined. One of: • **"system"** – Let the system determine the fenced book values. • **"user"** – Let the user determine the fenced book values.
`fenced-book-list`	Array of fenced-book-data objects	Optional	List of fenced-book-data objects. **Note:** If passed in with the request, the minimum number of members in this list is **1**, while the maximum number of members in this list is **5**. **Note:** Each member must have a unique **pu-mcm-size**. If more than one **fenced-book-data** object is sent in the request, none of their **pu-mcm-size** can be **0**.

Name	Type	Rqd/ Opt	Description
`partition-profile-names`	Array of String (1-255)	Optional	Specifies the order in which the logical partitions will be activated. The minimum value for the **partition-profile-names** property is 1; the maximum is the value of the CPC Object's Data Model **maximum-partitions** property. **Note:** Valid partition entries to this must be alphanumeric, with a length of 1 to 8 characters.

Table 530. fenced-book-data nested object properties

Name	Type	Rqd/Opt	Description
`pu-mcm-size`	Integer (0-255)	Required	The number of PU (processing units) on the card that is being fenced
`num-cp-fenced`	Integer (0-255)	Optional	The number of central processors to use when the book is fenced.
`num-sap-fenced`	Integer (1-255)	Optional	The number of System Assist Processors (SAP) to use when this book is fenced.
`num-icf-fenced`	Integer (0-255)	Optional	The number of Internal Coupling Facility (ICF) processors to use when this book is fenced.
`num-ifl-fenced`	Integer (0-255)	Optional	Number of Integrated Facility for Linux (IFL) processors to use when this book is fenced.
`num-ziip-fenced`	Integer (0-255)	Optional	Number of IBM z Integrated Information Processors (zIIP) to use when this book is fenced.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The element URI of the newly created Reset activation profile.

Description

The `Create Reset Activation Profile` operation creates a new reset profile on the Support Element. This new profile will be created with the name indicated in the **name** field and will be created using the values from the DEFAULT Reset profile, unless the **copy-name** field is provided in the request, in which case the profile will be created with the values from the **copy-name** identified profile as initial values. The **copy-name** field, if provided, must contain a valid existing profile name of the same type. Users of this operation can override specific properties by specifying them in the request body.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. On successful execution, the Reset Activation Profile is created, and a status code 201 (Created) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned, and the response body is provided as described in "Response body contents" on page 1287, and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The **name** value for this profile type already exists on the CPC with **object-id** *{cpc-id}*.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The object designated by the request URI does not support the requested operation.
	260	The activation profile name in the copy-name field does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	399	The operation cannot be performed as it would exceed the maximum number of Reset Activation profiles allowed on the CPC. Retry the operation after a Reset Activation profile is deleted from the CPC.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/reset-activation-profiles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
Request Body:
{
"profile-name":"TESTRES1",
"copy-name":"DEFAULT"
}
```

Figure 673. Create Reset Activation Profile: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91
cache-control: no-cache
date: Tue, 16 May 2023 01:35:07 GMT
content-type: application/json;charset=UTF-8
content-length: 99
Response Body:
{
"element-uri":"/api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/reset-activation-
profiles/TESTRES1"
}
```

Figure 674. Create Reset Activation Profile: Response

Delete Reset Activation Profile

The `Delete Reset Activation Profile` operation deletes a Reset activation profile, by profile name designated by *{reset-activation-profile-name}* from the Support Element. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}
```

URI variables:

Table 531.

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{reset-activation-profile-name}	Reset Activation Profile name.

Description

The `Delete Reset Activation Profile` operation deletes a Reset Activation profile from the SE. This profile will be deleted based on the *{reset-activation-profile-name}*.

The URI path must designate an existing Reset Activation Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{reset-activation-profile-name}*) does not designate an existing Reset Activation Profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/reset-activation-profiles/TESTRES1
x-api-session: 5hirq4lowb9dweimnmoedk5u1qk2ltlfo9f56xrs9kux6w6gzv
```

Figure 675. Delete Reset Activation Profile: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 16 May 2023 00:33:27 GMT

<No response body>
```

Figure 676. Delete Reset Activation Profile: Response

Inventory service data

Information about reset activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Reset Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular reset activation profile is included only if the API user has access permission to that object as described in the `Get Reset Activation Profile Properties` operation.

For each Reset Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Reset Activation Profile Properties" on page 1281. That is, the data provided is the same as would be provided if a `Get Reset Activation Profile Properties` operation were requested targeting this object.

Image activation profile

An Image activation profile is used by an Activate operation to activate a logical partition of a previously activated CPC.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Image activation profiles named "DEFAULT" or whose name begins with "0D0" do not have any certificates in them and cannot be assigned certificates. These are special image activation profiles that are firmware created and owned. [Updated by feature **create-delete-activation-profiles**]

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This element includes the following properties. Some properties have additional notes associated with them. Refer to the table notes at the end of this table.

Table 532. Image activation profile: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Image Activation Profile object, of the form `/api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}` where *{image-activation-profile-name}* is the value of the name property (Image Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object.
class	—	String	The class of an Image Activation Profile object is **"image-activation-profile"**.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
name	(pc)	String (0-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by *{cpc-id}*.
description	(w)	String (0-50)	The activation profile description
ipl-address[15]	(w)(pc)	String (0-5)	The hexadecimal address of an I/O device that provides access to the control program to be loaded. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Get: If this property contains an address, the address is either four- or five-digit (in hex) depending on its actual value and the value of **ipl-type**. Update: When this property is intended to be used for **ipl-type** of **"ipltype-nvmeload"** or **"ipltype-nvmedump"**, the input value shall be an empty string or a four-digit hexadecimal function ID (FID) of the NVMe device. If it's a FID, it will be right-justified and padded with zeros. The values shall be in the range "0000" to "FFFF". For any other **ipl-type**, the input value shall be an empty string or a five-digit hexadecimal address of the I/O device. If it's an address, it will be right-justified and padded with zeros. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". [Updated by feature **secure-boot-with-certificates**]
ipl-parameter	(w)(pc)	String (0-8)	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank and period. Three additional characters, (@, $, #) are also allowed when the **se-version** property of the associated CPC is **"2.14.0"** or later. On an Update, a non-empty string is left justified and right padded with blanks to 8 characters.
initial-processing-weight[1]	(w)(pc)	Integer	The relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** The Image Activation Profile does not represent a logical partition with at least one shared general purpose processor. **1-999** Represents the relative amount of shared general purpose processor resources initially allocated to the logical partition. Update: **1-999** Define the relative amount of shared general purpose processor resources allocated to the logical partition.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
initial-processing-weight-capped[1, 2, 3]	(w)(pc)	Boolean	Whether the initial processing weight for general purpose processors is a limit or a target. **True:** Indicates that the initial general purpose processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources. **False:** Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.
minimum-processing-weight[1]	(w)(pc)	Integer	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared general purpose processor resources allocated to the logical partition. The value must be less than or equal to the initial-processing-weight property.
maximum-processing-weight[1]	(w)(pc)	Integer	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Defines the maximum relative amount of shared general purpose processor resources allocated to the logical partition. Must be greater than or equal to the initial-processing-weight property.
absolute-general-purpose-capping[22]	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the general purpose processor.
workload-manager-enabled[4]	(w)(pc)	Boolean	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. **True:** Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. **False:** Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
defined-capacity	(w)	Integer	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed to by z/OS Workload Manager for the purpose of software pricing. **0:** No defined capacity is specified for this logical partition. **1-nnnn:** Represents the amount of defined capacity specified for this logical partition
ipl-type	(w)(pc)	String Enum	One of: • **"ipltype-scsi"** - This image activation profile is used to perform a SCSI list-directed OS load. • **"ipltype-scsidump"** - This image activation profile is used to perform a SCSI list-directed OS dump. • **"ipltype-nvmeload"** – This image activation profile is used to perform an NVMe list-directed OS load. • **"ipltype-nvmedump"** – This image activation profile is used to perform an NVMe list-directed OS dump. • **"ipltype-tape-load"** - This image activation profile is used to perform a tape Channel Command Word (CCW) OS load. • **"ipltype-tape-dump"** - This image activation profile is used to perform a tape Channel Command Word (CCW) OS dump. • **"ipltype-eckd-ccw-load"** - This image activation profile is used to perform an ECKD Channel Command Word OS load. • **"ipltype-eckd-ccw-dump"** - This image activation profile is used to perform an ECKD Channel Command Word OS dump. • **"ipltype-eckd-ld-load"** - This image activation profile is used to perform an ECKD list-directed OS load. • **"ipltype-eckd-ld-dump"** - This image activation profile is used to perform an ECKD list-directed OS dump. • **"ipltype-standard"** - This image activation profile is used to perform a Channel Command Word (CCW) standard load. This is present for compatibility, and the more specific **"ipltype-tape-load" "ipltype-tape-dump" "ipltype-eckd-ccw-load"** or **"ipltype-eckd-ccw-dump"** is preferred instead on CPCs with feature **secure-boot-with-certificates**. In that case, if a value of **ipltype-standard** is provided in an update, the value will be interpreted to one of the new values. **"ipltype-eckd-ccw-dump"** will be set if the user specifies the **store-status-indicator** to **true**, otherwise **"ipltype-eckd-ccw-load"** will be set. Any operation that previously worked with **"ipltype-standard"** will continue to work with this translation. For an Update operation, **"ipltype-nvmeload"** and **"ipltype-nvmedump"** are valid only when the associated SE version is 2.15.0 with suitable MCL bundle, or a later SE version, and **"ipltype-tape-load"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ccw-load"**, **"ipltype-eckd-ccw-dump"**, **"ipltype-eckd-ld-load"**, **"ipltype-eckd-ld-dump"** values are only valid on when the associated CPC has feature **secure-boot-with-certificates**. [Updated by feature **secure-boot-with-certificates**]
worldwide-port-name[15]	(w)	String (1-16)	Worldwide port name of the target SCSI device, used for a SCSI load or SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
disk-partition-id	(w)	Integer (0-30)	The disk partition number (also called the boot program selector) for the activation profile, used for a list-directed OS load or list-directed dump. When the **disk-partition-id-automatic** property is set to **true**, the value in this field is overridden and not used to complete the load. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
disk-partition-id-automatic	(w)	Boolean	Whether the value for **disk-partition-id** (also known as the boot program selector) used for a load should be determined automatically. When this field is set to **true**, it will override any value in **disk-partition-id**. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. This field is only permitted when the SE version of the owning CPC has feature **secure-boot-with-certificates**. [Added by feature **secure-boot-with-certificates**]
logical-unit-number[15]	(w)	String (1-16)	Logical unit number value for the activation profile, used for a SCSI load or SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.
boot-record-lba[15]	(w)	String (1-16)	Boot record logical block address for the activation profile, used for a SCSI load or SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, or **"ipltype-nvmedump"**.
os-specific-load-parameters	(w)	String (0-256)	Operating system-specific load parameters for the activation profile, used for a list-directed OS load or list-directed OS dump. On an Update, value is left-justified and right-padded with blanks to 256 characters. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
boot-record-location-cylinder	(w)	String (1-7)	The boot record location cylinder value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC has feature **secure-boot-with-certificates**. [Added by feature **secure-boot-with-certificates**]
boot-record-location-head	(w)	String (1)	The boot record location head value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC has feature **secure-boot-with-certificates**. [Added by feature **secure-boot-with-certificates**]
boot-record-location-record	(w)	String (0-2)	The boot record location record in hexadecimal. The record may not be set to "0" or "00". **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC has feature **secure-boot-with-certificates**. [Added by feature **secure-boot-with-certificates**]

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-record-location-use-volume-label	(w)	Boolean Optional when **load-type** is **"iplype-eckd-ld-load"**. Invalid otherwise	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** should be determined by the volume label. On an update, this value cannot be set to **true** if the **boot-record-location-cylinder**, **boot-record-location-head**, or **boot-record-location-record** is specified in the same request. If this value is **true**, it overrides the **boot-record-location-cylinder,boot-record-location-head**, and **boot-record-location-record** when a load is completed with this profile. Default: True if **boot-record-location** is not specified. False otherwise. [Added by feature **secure-boot-with-certificates**]
initial-aap-processing-weight[5, 21]	(w)	Integer	The relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition at activation. Get: **0** The Image Activation Profile does not represent a logical partition with at least one shared Application Assist Processor (zAAP) processor. **1-999** Represents the relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: **1-999** Define the relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
initial-aap-processing-weight-capped[3, 5, 6, 21]	(w)	Boolean	Whether the initial processing weight for Application Assist Processor (zAAP) processors is a limit or a target. **True:** Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Application Assist Processor (zAAP) processor resources, regardless of the availability of excess Application Assist Processor (zAAP) processor resources. **False:** Indicates that the initial Application Assist Processor (zAAP) processor processing weight for the logical partition is not capped. It represents the share of Application Assist Processor (zAAP) processor resources guaranteed to a logical partition when all Application Assist Processor (zAAP) processor resources are in use. Otherwise, when excess Application Assist Processor (zAAP) processor resources are available, the logical partition can use them if necessary.
minimum-aap-processing-weight[5, 21]	(w)	Integer	The minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: **0** No minimum value for the processing weight. **1-999** Define the minimum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
maximum-aap-processing-weight[5, 21]	(w)	Integer	The maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Application Assist Processor (zAAP) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Application Assist Processor (zAAP) processor resources allocated to the logical partition.
absolute-aap-capping[21, 22]	(w)	absolute-capping object	The amount of absolute capping applied to the Application Assist Processor (zAAP).
initial-ifl-processing-weight[7]	(w)(pc)	Integer	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition at activation. Get: **0** The Image Activation Profile does not represent a logical partition with at least one shared Integrated Facility for Linux (IFL) processor. **1-999** Represents the relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: **1-999** Define the relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
initial-ifl-processing-weight-capped[3, 7, 8]	(w)(pc)	Boolean	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. **True:** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources. **False:** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
minimum-ifl-processing-weight[7]	(w)(pc)	Integer	The minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
maximum-ifl-processing-weight[7]	(w)(pc)	Integer	The maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition.
absolute-ifl-capping[22]	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
initial-internal-cf-processing-weight[9]	(w)(pc)	Integer	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition at activation. Get: **0** The Image Activation Profile does not represent a logical partition with at least one shared Internal Coupling Facility (ICF) processor. **1-999** Represents the relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: **1-999** Define the relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
initial-internal-cf-processing-weight-capped[3, 9, 10]	(w)(pc)	Boolean	Indicates whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. **True:** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the Image associated with the logical partition is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. **False:** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the logical partition is not capped. It represents the share of Internal Coupling Facility (ICF) processor resources guaranteed to a logical partition when all Internal Coupling Facility (ICF) processor resources are in use. Otherwise, when excess Internal Coupling Facility (ICF) processor resources are available, the logical partition can use them if necessary.
minimum-internal-cf-processing-weight[9]	(w)(pc)	Integer	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition at activation. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: **1-999** Define the minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.
maximum-internal-cf-processing-weight[9]	(w)(pc)	Integer	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition.
absolute-icf-capping[22]	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
initial-ziip-processing-weight[11]	(w)(pc)	Integer	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition at activation. Get: **0** The Image Activation Profile does not represent a logical partition with at least one shared z Integrated Information Processors (zIIP) processor. **1-999** Represents the relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: **1-999** Define the relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
initial-ziip-processing-weight-capped[3, 11, 12]	(w)(pc)	Boolean	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. **True:** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. **False:** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.
minimum-ziip-processing-weight[11]	(w)(pc)	Integer	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no minimum value for the processing weight. **1-999** Represents the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: **0** There is no minimum value for the processing weight. **1-999** Define the minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
maximum-ziip-processing-weight[11]	(w)(pc)	Integer	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. Get: **0** There is no maximum value for the processing weight. **1-999** Represents the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources initially allocated to the logical partition. Update: **0** There is no maximum value for the processing weight. **1-999** Define the maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition.
absolute-ziip-capping[22]	(w)(pc)	absolute-capping object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor.
group-profile-uri	(w)	String/ URI	The canonical URI of the Group profile to be used for the logical partition activated by this profile, which provides the group capacity value. On a Get, a null object is returned if no Group profile is associated with this activation profile. On an Update, a null object indicates that no Group profile is to be associated with this activation profile.
load-at-activation	(w)	Boolean	If true, the logical partition will be loaded at the end of the activation.
secure-boot[25]	(w)(pc)	Boolean	If **true**, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the logical partition. The load will fail if the signatures do not match. Default: **false** **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
central-storage	(w)(pc)	Integer	Defines the amount of central storage, measured in megabytes (MB), to be allocated for the logical partition's exclusive use at activation. This value must be a multiple of the storage granularity value.
reserved-central-storage	(w)	Integer	Defines the amount of central storage, measured in megabytes (MB), dynamically reconfigurable to the logical partition after activation. This value must be a multiple of the storage granularity value.
expanded-storage	(w)(pc)	Integer	Defines the amount of expanded storage, measured in megabytes (MB), to be allocated for the logical partition's exclusive use at activation. This value must be a multiple of the storage granularity value.
reserved-expanded-storage	(w)	Integer	Defines the amount of expanded storage, measured in megabytes (MB), dynamically reconfigurable to the logical partition after activation. This value must be a multiple of the storage granularity value.
initial-vfm-storage[18]	(w)	Long	Defines the amount of Virtual Flash Memory (VFM) storage, measured in gigabytes (GB), to be allocated for the logical partition's exclusive use at activation. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC. This value must not be greater than the value specified in the **maximum-vfm-storage** property in this Image Activation Profile.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
maximum-vfm-storage[18]	(w)	Long	Defines the maximum amount of VFM storage, measured in gigabytes (GB), that can be allocated to the logical partition while it is running. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC. This value must not be smaller than the value specified in the **initial-vfm-storage** property in this Image Activation Profile.
processor-usage	(w)(pc)	String Enum	Defines how processors are allocated to the logical partition. One of the following values: • **"dedicated"** - all processor types in the logical partition are to be exclusively available to this specific logical partition. • **"shared"** - all processors types in the logical partition are to be shareable across logical partitions.
number-dedicated-general-purpose-processors[13]	(w)(pc)	Integer	Defines the number of general purpose processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-general-purpose-processors[13]	(w)	Integer	Defines the number of dedicated general purpose processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-aap-processors[13, 21]	(w)	Integer	Defines the number of Application Assist Processor (zAAP) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-aap-processors[13, 21]	(w)	Integer	Defines the number of dedicated Application Assist Processor (zAAP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-ifl-processors[13]	(w)(pc)	Integer	Defines the number of Integrated Facility for Linux (IFL) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-ifl-processors[13]	(w)	Integer	Defines the number of dedicated Integrated Facility for Linux (IFL) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-icf-processors[13]	(w)(pc)	Integer	Defines the number of Integrated Coupling Facility (ICF) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-icf-processors[13]	(w)	Integer	Defines the number of dedicated Integrated Coupling Facility (ICF) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-dedicated-ziip-processors[13]	(w)(pc)	Integer	Defines the number of z Integrated Information Processors (zIIP) processors to be allocated for the logical partition's exclusive use at activation.
number-reserved-dedicated-ziip-processors[13]	(w)	Integer	Defines the number of dedicated z Integrated Information Processors (zIIP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-general-purpose-processors[14]	(w)(pc)	Integer	Defines the number of shared general purpose processors to be allocated for the logical partition at activation.
number-reserved-shared-general-purpose-processors[14]	(w)	Integer	Defines the number of shared general purpose processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-aap-processors[14, 21]	(w)	Integer	Defines the number of shared Application Assist Processor (zAAP) processors to be allocated for the logical partition at activation.
number-reserved-shared-aap-processors[14, 21]	(w)	Integer	Defines the number of shared Application Assist Processor (zAAP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-ifl-processors[14]	(w)(pc)	Integer	Defines the number of shared Integrated Facility for Linux (IFL) processors to be allocated for the logical partition at activation.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
number-reserved-shared-ifl-processors[14]	(w)	Integer	Defines the number of shared Integrated Facility for Linux (IFL) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-icf-processors[14]	(w)(pc)	Integer	Defines the number of shared Integrated Coupling Facility (ICF) processors to be allocated for the logical partition at activation.
number-reserved-shared-icf-processors[14]	(w)	Integer	Defines the number of shared Integrated Coupling Facility (ICF) processors to be reserved for the logical partition, which can be dynamically configured after activation.
number-shared-ziip-processors[14]	(w)(pc)	Integer	Defines the number of shared z Integrated Information Processors (zIIP) processors to be allocated for the logical partition at activation.
number-reserved-shared-ziip-processors[14]	(w)	Integer	Defines the number of shared z Integrated Information Processors (zIIP) processors to be reserved for the logical partition, which can be dynamically configured after activation.
basic-cpu-counter-authorization-control	(w)	Boolean	If true, the basic CPU counter facility for the logical partition is enabled.
problem-state-cpu-counter-authorization-control	(w)	Boolean	If true, the problem state CPU counter facility for the logical partition is enabled.
crypto-activity-cpu-counter-authorization-control	(w)	Boolean	If true, the crypto activity CPU counter facility for the logical partition is enabled.
extended-cpu-counter-authorization-control	(w)	Boolean	If true, the extended CPU counter facility for the logical partition is enabled.
coprocessor-cpu-counter-authorization-control	(w)	Boolean	If true, the coprocessor group CPU counter facility for the logical partition is enabled.
basic-cpu-sampling-authorization-control	(w)	Boolean	If true, the basic CPU sampling facility for the logical partition is enabled.
permit-aes-key-import-functions[24]	(w)	Boolean	If true, importing of Advanced Encryption Standard (AES) keys for the logical partition is enabled.
permit-des-key-import-functions[24]	(w)	Boolean	If true, importing of Data Encryption Standard (DES) keys for the logical partition is enabled.
permit-ecc-key-import-functions[23, 24]	(w)	Boolean	If true, importing of Elliptic Curve Cryptography (ECC) keys for the logical partition is enabled.
liccc-validation-enabled	(w)	Boolean	If true, ensure that the image profile data conforms to the current maximum Licensed Internal Code Configuration Control (LICCC) configuration.
assigned-crypto-domains	—	Array of assigned-crypto-domain objects	Array of assigned-crypto-domain objects, as described in Table 533 on page 1307. Specifies the **"control"** domain and **"control and usage"** domain indexes to be assigned to the logical partition once activated.
assigned-cryptos	—	Array of assigned-crypto objects	Array of assigned-crypto objects, as described in Table 534 on page 1307. Specifies the cryptographic **"candidate"** list and cryptographic **"candidate and online"** list settings to be assigned to the logical partition once activated.
global-performance-data-authorization-control	(w)	Boolean	If true, the logical partition can be used to view the processing unit activity data for all other logical partitions activated on the same CPC.
io-configuration-authorization-control	(w)	Boolean	If true, the logical partition can be used to read and write any Input/Output Configuration Data Set (IOCDS) in the configuration.
cross-partition-authority-authorization-control	—	Boolean	If true, the logical partition can be used to issue control program instructions that reset or deactivate other logical partitions.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
logical-partition-isolation-control	(w)	Boolean	If true, reconfigurable channel paths assigned to the logical partition are reserved for its exclusive use.
operating-mode	(w)(pc)	String Enum	The operating mode for the logical partition: • **"general"** • **"esa390"** • **"esa390-tpf"** • **"coupling-facility"** • **"linux-only"** • **"zvm"** • **"zaware"** • **"ssc"** For an Update operation, **"ssc"** is only valid when the associated Support Element is at version 2.13.1 or later and **"zaware"** is only valid when the associated Support Element is at version 2.13.0 or earlier. For an Update operation, **"general"** is only valid when the associated Support Element is at version 2.14.0 or later and **"esa390"** or **"esa390-tpf"** is only valid when the associated Support Element is at version 2.13.1 or earlier.
clock-type	(w)	String Enum	One of: • **"standard"** - Set the logical partition's clock is set to the same time set for the CPC's time source. • **"lpar"** - Set the logical partition's clock using an offset from the External Time Source's time of day.
time-offset-days	(w)	Integer (0-999)	The number of days the logical partition's clock is to be offset from the External Time Source's time of day.
time-offset-hours	(w)	Integer (0-23)	The number of hours the logical partition's clock is to be offset from the External Time Source's time of day.
time-offset-minutes	(w)	Integer Enum	The number of minutes the logical partition's clock is to be offset from the External Time Source's time of day. Allowable values are 0, 15, 30 or 45.
time-offset-increase-decrease	(w)	String Enum	One of: • **"increase"** - Set the logical partition's clock ahead of the External Time Source's time of day. • **"decrease"** - Set the logical partition's clock back from the External Time Source's time of day.
zaware-host-name[19]	(w)	String (1-64)	The IBM zAware host name. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and colon(:)
zaware-master-userid[19]	(w)	String (1-32)	The IBM zAware master userid. Valid characters are: a-z,A-Z,0-9, period(.), minus(-) and underscore (_)
zaware-master-pw[19]	(wo)	String (8-256)	The IBM zAware master password. Valid characters are: a-z,A-Z,0-9 and ! @#$%^&*()_+{}|<>?-= This property is not returned on a Get request, it can only be specified on an Update request.
zaware-network-info[19]	(w)	Array of zaware-network objects	The set of networks available to IBM zAware. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
zaware-gateway-info[16]	(w)	ip-info object	The default gateway IP address information. A null object indicates no default gateway IP address is specified.
zaware-dns-info[16]	(w)	Array of ip-info objects	The DNS IP address information. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.

Table 532. Image activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ssc-host-name[20]	(w)	String (1-64)	The Secure Service Container host name. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-) and colon(:).
ssc-master-userid[20]	(w)	String (1-32)	The Secure Service Container master user ID. Valid characters are: a-z, A-Z, 0-9, period(.), minus(-) and underscore(_).
ssc-master-pw[20]	(wo)	String (8-256)	The Secure Service Container master password. Valid characters are: a-z, A-Z, 0-9, and !@#$%^&*()_+{}|<>?-=. This property is not returned on a Get request; it can be specified on an Update request.
ssc-network-info[20]	(w)	Array of ssc-network objects	The set of networks available to the Secure Service Container. A minimum of 1 network and a maximum of 100 networks are permitted. On an Update request, this property fully replaces the existing set.
ssc-gateway-info[17]	(w)	ip-info objects	The IPv4 default gateway IP address information for the Secure Service Container. A null object indicates no IPv4 default gateway IP address is specified.
ssc-gateway-ipv6-info[17, 18]	(w)	ip-info objects	The IPv6 default gateway IP address information for the Secure Service Container. A null object indicates no IPv6 default gateway IP address is specified.
ssc-dns-info[17]	(w)	Array of ip-info objects	The DNS IP address information for the Secure Service Container. A minimum of 0 entries and a maximum of 2 entries are permitted. On an Update request, this property fully replaces the existing set.
ssc-boot-selection[20]	(w)	String Enum	Indicates whether to run the Secure Service Container appliance installer or the Secure Service Container appliance itself. One of: • **"installer"** - Boot the Secure Service Container appliance installer to install the Secure Service Container appliance and then start it. • **"appliance"** - Reload the most recently installed Secure Service Container appliance and resume its execution from where it was when the image was deactivated.
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
partition-identifier	(w)	String (2)	The two-digit hexadecimal partition identifier to be used for the logical partition.
assigned-certificate-uris	(c)(pc)	Array of String/ URI	Array of URIs referring to the certificates that are assigned to this image activation profile, or an empty array if there are no assigned certificates. [Added by feature **secure-boot-with-certificates**]

Table 532. Image activation profile: properties (continued)

Name	Qualifier	Type	Description

Notes:

1. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared general purpose processor.
2. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition or the Image Activation Profile does not represent a logical partition with at least one shared general purpose processor.
3. This property and the **workload-manager-enabled** property are mutually exclusive and cannot both be enabled at the same time. Therefore in order to enable this property it might be necessary to first disable the **workload-manager-enabled** property.
4. This property and the various capping properties are mutually exclusive and cannot be enabled at the same time. Therefore, in order to enable this property it may be necessary to first disable any capping property that is currently enabled.
5. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Application Assist Processor (zAAP) processor.
6. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Application Assist Processor (zAAP) processor.
7. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Integrated Facility for Linux (IFL) processor.
8. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Integrated Facility for Linux (IFL) processor.
9. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared Internal Coupling Facility (ICF) processor.
10. The value returned for a Get request is always false when the Image Activation Profile does not represent a logical partition with at least one shared Internal Coupling Facility (ICF) processor.
11. An Update of this property is only valid for an Image Activation Profile that represents a logical partition with at least one shared z Integrated Information Processors (zIIP) processor.
12. The value returned for a GET request is always false when the Image Activation profile does not represent a logical partition with at least one shared z Integrated Information Processors (zIIP) processor.
13. The value of this property is a null object if the processor-usage property is **"shared"**
14. The value of this property is a null object if the processor-usage property is **"dedicated"**
15. An Update request accepts any mixture of [a-f,A-F,0-9], however the original string value is not saved and a subsequent Get request may not return the exact same set of lower/upper case letters.
16. On a Get request, this property is returned only when **operating-mode** is **"zaware"**. On an Update request, this property can be specified only when **operating-mode** is **"zaware"**.
17. On a Get request, this property is returned only when **operating-mode** is **"ssc"**. On an Update request, this property can be specified only when **operating-mode** is **"ssc"**.
18. On a Get request, this property is returned only when the SE version is 2.14.0 or later. On an Update request, this property is allowed only when the SE version is 2.14.0 or later.
19. On an Update request, this property can be specified only when **operating-mode** is set to **"zaware"** in the same request.
20. On an Update request, this property can be specified only when **operating-mode** is set to **"ssc"** in the same request.
21. On a Get request, this property is not returned when the SE version is 2.13.0 or later. On an Update request, this property is ignored when the SE version is 2.13.0 or later.
22. On a Get request, this property is not returned for a profile whose **processor-usage** property is **"dedicated"**. On an Update request, this property is not allowed for a profile whose **processor-usage** property is **"dedicated"**.
23. On a Get Request, this property is returned only when the SE version is 2.15.0 or later. On an Update request, this property is only allowed when the SE version is 2.15.0 or later.
24. Updating this property will have no effect if the SE does not have CP Assist for Cryptographic functions (CPACF) enabled.
25. On a Get Request, this property is returned only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.

Each nested assigned-crypto-domain object contains the following properties:

Table 533. assigned-crypto-domain nested object		
Field Name	**Type**	**Description**
domain-index	Integer	Index value that identifies the domain to which this configuration applies. Minimum index is 0, maximum index depends on the CPC model.
access-mode	String Enum	Specifies the way in which the partition can use this domain. Valid values are: • **"control"** - The partition can load cryptographic keys into the domain, but it may not use the domain to perform cryptographic operations. • **"control-usage"** - The partition can load cryptographic keys to the domain, and it can use the domain to perform cryptographic operations.

Each nested assigned-crypto object contains the following properties:

Table 534. assigned-crypto nested object		
Field Name	**Type**	**Description**
number	Integer	Identification number of the crypto adapter to be assigned during partition activation.
activation-type	String Enum	Specifies the way in which the crypto will be handled during partition activation. Valid values are: • **"candidate"** - The crypto adapter will be assigned to the logical partition during partition activation. • **"candidate-online"** - The crypto adapter will be assigned to the logical partition during partition activation, then brought online during this partition activation.

List Image Activation Profiles

The `List Image Activation Profiles` operation lists the Image Activation Profiles for the associated CPC object. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/image-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
`name`	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.
`additional-properties`	List of String Enum	Optional	A list of properties to be included in the response in addition to the default properties. This is a list of comma-separated strings where each string is a property name defined in the Image Activation Profile's “Data model” on page 1291. [Added by feature **secure-boot-with-certificates**]

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`image-activation-profiles`	Array of image-actprof-info objects	Array of nested objects (described in the following table).

Each image-actprof-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path of the Image Activation Profile object.
`name`	String	The name of the Image Activation Profile.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Image Activation Profiles associated with a particular CPC.

If the **name** query parameter is specified, the returned list is limited to those Image Activation Profiles that have a name property matching the specified filter pattern. If the **name** parameter is omitted, this filtering is not done.

If the additional-properties parameter is specified, additional properties are included in the returned list. The properties to be included is a list of comma-separated strings where each string is a property name defined in the Image Activation Profile's data model. If the additional-properties parameter is omitted, no such properties will be included. [Updated by feature **secure-boot-with-certificates**]

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1308.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to the same named logical partition of that CPC, for each object to be included in the response.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1308.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 677. List Image Activation Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 506
{
   "image-activation-profiles": [
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
           APIVM1",
         "name": "APIVM1"
      },
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
           DEFAULT",
         "name": "DEFAULT"
      },
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-
profiles/
           ZOS1",
         "name": "ZOS1"
      }
   ]
}
```

Figure 678. List Image Activation Profiles: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI through the `List CPC Objects` operation. Instead, such a user may use either the `List Permitted Logical Partitions` operation or the `Get Inventory` operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Get Image Activation Profile Properties

The `Get Image Activation Profile Properties` operation retrieves the properties of a single Image Activation Profile designated by *{image-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}
```

URI variables:

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{image-activation-profile-name}	Image Activation Profile name

Query parameters:

Name	Type	Rqd/Opt	Description
`cached-acceptable`	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**. The default is **false**.
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the object's data model.

Response body contents

On successful completion, the response body provides the current values of the properties for the Image Activation Profile as defined in the "Data model" on page 1291.

Description

The URI path must designate an existing Image Activation Profile and the API user must have object-access permission to the associated CPC object. If either of these conditions is not met, HTTP status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some of this object's property values are periodically fetched from the Support Element and cached for quick access by the APIs. Due to the nature of this caching support, the cached value of a property may differ from the actual value at any point in time. While the cache is kept reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Image Activation Profile object.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to the same named logical partition of that CPC.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1311.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
	260	The activation profile name in the URI (*{image-activation-profile-name}*) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles/ZOS HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 679. Get Image Activation Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:18 GMT
content-type: application/json;charset=UTF-8
content-length: 3360
{
   “assigned-cryptos”:[
     {
       “number”:”32”,
       “activation-type”:”candidate-online”
     },
     {
       “number”:”59”,
       “activation-type”:”candidate”
     }
   ],
   “assigned-crypto-domains”:[
     {
       “domain-index”:”6”,
       “access-mode”:”control-usage”
     },
     {
       “domain-index”:”25”,
       “access-mode”:”control”
     },
     {
       “domain-index”:”44”,
       “access-mode”:”control-usage”
     }
   ],
   "basic-cpu-counter-authorization-control": false,
   "basic-cpu-sampling-authorization-control": false,
   "boot-record-lba": "0",
   "central-storage": 4096,
   "class": "image-activation-profile",
   "clock-type": "standard",
   "coprocessor-cpu-counter-authorization-control": false,
   "cross-partition-authority-authorization-control": false,
   "crypto-activity-cpu-counter-authorization-control": false,
   "defined-capacity": 0,
   "description": "This is the ZOS Image profile.",
   "disk-partition-id": 0,
   "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/image-activation-profiles/
     ZOS",
   "expanded-storage": 0,
   "extended-cpu-counter-authorization-control": false,
   "global-performance-data-authorization-control": true,
   "group-profile-uri": null,
   "initial-aap-processing-weight": 0,
   "initial-aap-processing-weight-capped": false,
   "initial-ifl-processing-weight": 0,
   "initial-ifl-processing-weight-capped": false,
   "initial-internal-cf-processing-weight": 0,
   "initial-internal-cf-processing-weight-capped": false,
   "initial-processing-weight": 44,
   "initial-processing-weight-capped": false,
   "initial-vfm-storage": 64,
   "initial-ziip-processing-weight": 0,
   "initial-ziip-processing-weight-capped": false,
   "io-configuration-authorization-control": true,
   "ipl-address": "00000",
   "ipl-parameter": "        ",
   "ipl-type": "ipltype-standard",
   "liccc-validation-enabled": true,
   "load-at-activation": false,
   "logical-partition-isolation-control": false,
   "logical-unit-number": "0",
```

Figure 680. Get Image Activation Profile Properties: Response (Part 1)

```
    "maximum-aap-processing-weight": 0,
    "maximum-ifl-processing-weight": 0,
    "maximum-internal-cf-processing-weight": 0,
    "maximum-processing-weight": 44,
    "maximum-vfm-storage": 512,
    "absolute-aap-capping": {"type": "processors", "value": 67.95},
    "absolute-icf-capping": {"type": "none"},
    "absolute-ifl-capping": {"type": "none"},
    "absolute-general-purpose-capping": {"type": "processors", "value": 3.03},
    "absolute-ziip-capping": {"type": "processors", "value": 95.95},
    "maximum-ziip-processing-weight": 0,
    "minimum-aap-processing-weight": 0,
    "minimum-ifl-processing-weight": 0,
    "minimum-internal-cf-processing-weight": 0,
    "minimum-processing-weight": 44,
    "minimum-ziip-processing-weight": 0,
    "name": "ZOS",
    "number-dedicated-aap-processors": null,
    "number-dedicated-general-purpose-processors": null,
    "number-dedicated-icf-processors": null,
    "number-dedicated-ifl-processors": null,
    "number-dedicated-ziip-processors": null,
    "number-reserved-dedicated-aap-processors": null,
    "number-reserved-dedicated-general-purpose-processors": null,
    "number-reserved-dedicated-icf-processors": null,
    "number-reserved-dedicated-ifl-processors": null,
    "number-reserved-dedicated-ziip-processors": null,
    "number-reserved-shared-aap-processors": 0,
    "number-reserved-shared-general-purpose-processors": 0,
    "number-reserved-shared-icf-processors": 0,
    "number-reserved-shared-ifl-processors": 0,
    "number-reserved-shared-ziip-processors": 0,
    "number-shared-aap-processors": 0,
    "number-shared-general-purpose-processors": 1,
    "number-shared-icf-processors": 0,
    "number-shared-ifl-processors": 0,
    "number-shared-ziip-processors": 0,
    "operating-mode": "esa390",
    "os-specific-load-parameters": " ",
    "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
    "permit-aes-key-import-functions": true,
    "permit-des-key-import-functions": true,
    "permit-ecc-key-import-functions": true,
    "problem-state-cpu-counter-authorization-control": false,
    "processor-usage": "shared",
    "reserved-central-storage": 0,
    "reserved-expanded-storage": 0,
    "secure-boot": true,
    "time-offset-days": 0,
    "time-offset-hours": 0,
    "time-offset-increase-decrease": "decrease",
    "time-offset-minutes": 0,
    "workload-manager-enabled": false,
    "worldwide-port-name": "0"
}
```


Figure 681. Get Image Activation Profile Properties: Response (Part 2)

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI through the `List CPC Objects` operation. Instead, such a user may use either the `List Permitted Logical Partitions` operation or the `Get Inventory` operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Update Image Activation Profile Properties

The `Update Image Activation Profile Properties` operation updates one or more writable properties of the Image Activation Profile designated by *{image-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}
```

URI variables:

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{image-activation-profile-name}	Image Activation Profile name

Request body contents

The request body is expected to contain one or more field names representing writable Image Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Image Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

To allow the API user to prepare the profile for future use, the input value for a field is not validated against the values of other fields in the profile, unless specified otherwise. This implies that, when the input value of a field is not applicable or is conflicting with the value of another field, the operation still allows the update, and the new value will be saved in the profile. It is up to the user to validate the content of the profile before it is actually being used.

On successful execution, the value of each corresponding property of the Image Activation Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Customize/Delete Activation Profiles** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
	306	One or more of the provided update values would result in an illegal state involving the **operating-mode** property.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission.
	260	The activation profile name in the URI (*{image-activation-profile-name}*) does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Assign Certificate to Image Activation Profile

The `Assign Certificate to Image Activation Profile` operation assigns a certificate of **type "secure-boot"** to an Image activation profile. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}/operations/
assign-certificate
```

URI variables:

Name	Type	Description
cpc-id	String	The Object ID of the CPC object.
image-activation-profile-name	String	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by *{cpc-id}*.

Request body contents

The request body is a JSON object with the following field:

Table 535.

Field name	Type	Rqd/ Opt	Description
`certificate-uri`	String/ URI	Required	The URI of the certificate to be assigned.

Description

This operation assigns a secure boot certificate to an image activation profile.

If the Image activation profile specified cannot be assigned certificates, because it is a system defined Image Activation Profile, a 400 (Bad Request) status code is returned. A 404 (Not Found) status code is returned if the request does not designate an existing Image Activation Profile, CPC, or Certificate, or if the API user does not have object-access permission to the object. If the API user doesn't have action/task permission to the **Assign Secure Boot Certificates** task, 403 (Forbidden) status code is returned. If the Certificate object is currently assigned to the Image Activation Profile, if assigning the Certificate would exceed the Certificate limit of 20 per profile, or if attempting to assign to an unmanaged CPC, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*.
  - Object-access permission to the Certificate object whose **object-id** is *{certificate-id}*.
  - Action/task permission for the **Assign Secure Boot Certificates** task.
- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	374	The image activation profile specified cannot be assigned/ unassigned certificates.

HTTP error status code	Reason code	Description
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission
	4	The object designated by the request URI does not support the requested operation.
	260	The operation cannot be performed because the Image Activation Profile does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC identified by the request URI is an unmanaged CPC, which is not supported by this operation.
	372	The operation could not be performed because this certificate would exceed limit of 20 certificates per image activation profile.
	373	The operation cannot be performed because the certificate has already been assigned to this image activation profile.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4/image-activation-profiles/LP11/operations/
assign-certificate HTTP/1.1
x-api-session: 1sdt4yamt1f8horxk2an1xiua2tnaz2368a0a9o7sll8f8y7m
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/ab07f6ca-402f-11ed-ab57-fa163e6f7e7e"
}
```

Figure 682. Assign Certificate to Image Activation Profile: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 11 Oct 2022 15:24:47 GMT

<No response body>
```

Figure 683. Assign Certificate to Image Activation Profile: Response

Unassign Certificate from Image Activation Profile

The `Unassign Certificate from Image Activation Profile` operation unassigns a certificate of **type "secure-boot"** from an Image activation profile. This operation is supported using the BCPii interface. [Added by feature **secure-boot-with-certificates**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}/operations/
unassign-certificate
```

URI variables:

Name	Type	Description
cpc-id	String	The Object ID of the CPC object.
image-activation-profile-name	String	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by *{cpc-id}*.

Request body contents

The request body is a JSON object with the following field:

Field name	Type	Description
`certificate-uri`	String/ URI	The URI of the certificate to be unassigned.

Description

This operation unassigns a secure boot certificate from an image activation profile.

If certificates cannot be unassigned from the Image Activation Profile specified because it is a system defined Image Activation Profile, a 400 (Bad Request) status code is returned. A 404 (Not Found) status code is returned if the request does not designate an existing Image Activation Profile, CPC, or Certificate, or if the API user does not have object-access permission to the object. If the API user does not have action/task permission to the **Assign Secure Boot Certificates** task, 403 (Forbidden) status code is returned. If the Certificate object is not currently assigned to the Image Activation Profile, or if attempting to unassign from an unmanaged CPC, a 409 (Conflict) status code is returned. A 503 (Service Unavailable) status code is returned if the Console is not communicating with the CPC.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*.

- Object-access permission to the certificate object whose **object-id** is *{certificate-id}*.
- Action/task permission for the **Assign Secure Boot Certificates** task.

- For the BCPii interface the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	374	The image activation profile specified cannot be assigned/unassigned certificates.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	2	A URI in the request body does not designate an existing resource of the expected type, or designates a resource for which the API user does not have object-access permission
	4	The object designated by the request URI does not support the requested operation.
	260	The operation cannot be performed because the Image Activation Profile does not exist.
409 (Conflict)	329	The operation cannot be performed because the CPC identified by the request URI is an unmanaged CPC, which is not supported by this operation.
	370	The operation cannot be performed because the certificate is not assigned to this image activation profile.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/bab1c46f-17ca-3e5b-b93b-2669b2f344a4/image-activation-profiles/LP11/operations/
unassign-certificate HTTP/1.1
x-api-session: 5tga5xqrq9bd85qt2mvtw3qeksdf4i4djo2esnsnq547qdqmng
Content-Type: application/json
Content-Length: 77
{
   "certificate-uri":"/api/certificates/ab07f6ca-402f-11ed-ab57-fa163e6f7e7e"
}
```

Figure 684. Unassign Certificate from Image Activation Profile: Request

```
204
Server: Hardware management console API web server / 2.0
Cache-control: no-cache
Date: Tue, 11 Oct 2022 15:25:15 GMT

<No response body>
```

Figure 685. Unassign Certificate from Image Activation Profile: Response

Create Image Activation Profile

The `Create Image Activation Profile` operation creates a new Image activation profile. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/image-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

Name	Type	Rqd/ Opt	Description
`name`	String (1-8)	Required	The Image activation profile name, which uniquely identifies this profile within the set of Image activation profiles for the CPC object designated by *{cpc-id}*.
`copy-name`	String (1-8)	Optional	The name of an existing Image activation profile on the CPC object designated by *{cpc-id}*. If this field is provided, the name must be a valid name of an existing profile, which will then be loaded as the initial values for the Create operation. If this field is not sent in the request, the new image profile will use the fields from the DEFAULT image profile as the initial values for the created profile.
`description`	String (0-50)	Optional	The image profile description of the profile to be created.
`partition-identifier`	String (2)	Optional	The two-digit hexadecimal partition identifier to be used for the logical partition.

Name	Type	Rqd/ Opt	Description
`operating-mode`	String Enum	Optional	The operating mode for the logical partition: • **"general"** • **"coupling-facility"** • **"linux-only"** • **"zvm"** • **"ssc"**
`clock-type`	String Enum	Optional	One of: • **"standard"** - Set the logical partition's clock to the same time set for the CPC's time source. • **"lpar"** - Set the logical partition's clock using an offset from the External Time Source's time of day
`liccc-validation-enabled`	Boolean	Optional	If true, ensure that the image profile data conforms to the current maximum Licensed Internal Code Configuration Control (LICCC) configuration.
`group-profile-uri`	String/ URI	Optional	The name of the Group Profile associated with the logical partition.
`global-performance-data-authorization-control`	Boolean	Optional	If **true**, the logical partition can be used to view the processing unit activity data for all other logical partitions activated on the same CPC.
`io-configuration-authorization-control`	Boolean	Optional	If **true**, the logical partition can be used to read and write any Input/Output Configuration Data Set (IOCDS) in the configuration.
`cross-partition-authority-authorization-control`	Boolean	Optional	If **true**, the logical partition can be used to issue control program instructions that reset or deactivate other logical partitions.
`logical-partition-isolation-control`	Boolean	Optional	If **true**, reconfigurable channel paths assigned to the logical partition are reserved for its exclusive use.
`security-bcpii-send-commands`	Boolean	Optional	If **true**, configure the logical partition to allow the sending of commands through BCPii.
`security-bcpii-receive-commands`	Boolean	Optional	If **true**, configure the logical partition to allow the receiving of BCPii commands

Name	Type	Rqd/ Opt	Description
`security-bcpii-receive-selection`	String Enum	Optional	Determine the BCPii commands this image is allowed to receive. This value is only valid if **security-bcpii-receive-commands** is **True**. One of: • **"all"**– all BCPii commands are allowed to be received. • **"bcpii-list"** – only the BCPii commands in the list are allowed to be received.
`security-bcpii-receive-partition-list`	Array of String	Optional	The list of specified system/partitions from which the image, when configured to allow, can receive BCPii commands. The format for each String entry to the Array is as follows: *Netid*: 1-8 alphanumeric characters *System*: 1-8 alphanumeric characters *Partition*: 1-8 alphanumeric characters Example: NNNNNNNN.SSSSSSSS.PPPPPPPP
`basic-cpu-counter-authorization-control`	Boolean	Optional	If **true**, the basic CPU counter facility for the logical partition is enabled.
`problem-state-cpu-counter-authorization-control`	Boolean	Optional	If **true**, the problem state CPU counter facility for the logical partition is enabled
`crypto-activity-cpu-counter-authorization-control`	Boolean	Optional	If **true**, the crypto activity CPU counter facility for the logical partition is enabled.
`extended-cpu-counter-authorization-control`	Boolean	Optional	If **true**, the extended CPU counter facility for the logical partition is enabled
`basic-cpu-sampling-authorization-control`	Boolean	Optional	If **true**, the basic CPU sampling facility for the logical partition is enabled.
`diagnostic-sampling-authorization-control`	Boolean	Optional	If **true**, the diagnostic sampling authorization control is enabled.
`permit-des-key-import-functions`	Boolean	Optional	If **true**, importing of Data Encryption Standard (DES) keys for the logical partition is enabled.
`permit-aes-key-import-functions`	Boolean	Optional	If **true**, importing of Advanced Encryption Standard (AES) keys for the logical partition is enabled.

Name	Type	Rqd/ Opt	Description
`permit-ecc-key-import-functions`	Boolean	Optional	If **true**, importing of Elliptic Curve Cryptography (ECC) keys for the logical partition is enabled.
`central-storage`	Integer	Optional	Defines the amount of central storage, measured in megabytes (MB), to be allocated for the logical partition's exclusive use at activation. This value must be a multiple of the storage granularity value.
`reserved-central-storage`	Integer	Optional	Defines the amount of central storage, measured in megabytes (MB), dynamically reconfigurable to the logical partition after activation. This value must be a multiple of the storage granularity value.
`user-specified-central-storage-origin`	Boolean	Optional	If true, enable user specification of the central storage origin.
`central-storage-origin`	Long	Optional	The central storage origin. This is only used if user-specified-central-storage-origin is **true**.
`initial-vfm-storage`	Long	Optional	Defines the amount of Virtual Flash Memory (VFM) storage, measured in gigabytes (GB), to be allocated for the logical partition's exclusive use at activation. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC. This value must not be greater than the value specified in the **maximum-vfm-storage** property in this Image Activation Profile.
`maximum-vfm-storage`	Long	Optional	Defines the maximum amount of VFM storage, measured in gigabytes (GB), that can be allocated to the logical partition while it is running. The valid range is 0 to the value indicated on the **storage-vfm-total** property in a multiple of the value indicated on the **storage-vfm-increment-size** property for the associated CPC. This value must not be smaller than the value specified in the **initial-vfm-storage** property in this Image Activation Profile.
`minimum-io-priority-queueing`	Integer (0-255)	Optional	The minimum I/O priority queueing. **Note:** This value must be greater than or equal to 0 and less than the **maximum-io-priority-queueing** value.
`maximum-io-priority-queueing`	Integer (0-255)	Optional	The maximum I/O priority queueing. **Note:** This number must be greater than or equal to **minimum-io-priority-queueing**.
`logical-cluster-name`	String (0-8)	Optional	The CP management cluster name.

Name	Type	Rqd/ Opt	Description
`defined-capacity`	Integer	Optional	The defined capacity expressed in terms of Millions of Service Units (MSU)s per hour. MSU is a measure of processor resource consumption. The amount of MSUs a logical partition consumes is dependent on the model, the number of logical processors available to the partition, and the amount of time the logical partition is dispatched. The defined capacity value specifies how much capacity the logical partition is to be managed by z/OS Workload Manager for the purpose of software pricing. **0** No defined capacity is specified for this logical partition **1-nnnn** Represents the amount of defined capacity specified for this logical partition
`load-at-activation`	Boolean	Optional	If **true**, the logical partition will be loaded at the end of the activation.
`load-timeout`	Integer (60-600)	Optional	Amount of time, in seconds, to wait for the Load to complete
`store-status-indicator`	Boolean	Optional	Whether status should be stored before performing the Load (**true**) or not stored (**false**).

Name	Type	Rqd/ Opt	Description
`ipl-type`	String Enum	Optional	One of: • **"ipltype-scsi"** - This image activation profile is used to perform a SCSI list-directed OS load. • **"ipltype-scsidump"** - This image activation profile is used to perform a SCSI list-directed dump. • **"ipltype-nvmeload"** – This image activation profile is used to perform an NVMe list-directed OS load. • **"ipltype-nvmedump"** – This image activation profile is used to perform an NVMe list-directed dump. • **"ipltype-tape-load"** - This image activation profile is used to perform a tape Channel Command Word (CCW) OS load • **"ipltype-tape-dump"** - This image activation profile is used to perform a tape Channel Command Word (CCW) dump • **"ipltype-eckd-ccw-load"** - This image activation profile is used to perform an ECKD Channel Command Word OS load. • **"ipltype-eckd-ccw-dump"** - This image activation profile is used to perform an ECKD Channel Command Word dump • **"ipltype-eckd-ld-load"** - This image activation profile is used to perform an ECKD list-directed OS load • **"ipltype-eckd-ld-dump"** - This image activation profile is used to perform an ECKD list-directed dump • **"ipltype-standard"** - This image activation profile is used to perform a Channel Command Word (CCW) standard load. This is present for compatibility, and the more specific **"ipltype-tape-load" "ipltype-tape-dump" "ipltype-eckd-ccw-load"** or **"ipltype-eckd-ccw-dump"** is preferred instead on when the associated SE version is 2.16.0 and a sufficient MCL bundle. • In that case, if a value of **ipltype-standard** is provided in a create, the value will be interpreted to one of the new values. **"ipltype-eckd-ccw-dump"** will be set if the user specifies the **store-status-indicator** to **true**, otherwise **"ipltype-eckd-ccw-load"** will be set. Any operation that previously worked with **"ipltype-standard"** will continue to work with this translation. For a Create operation, **"ipltype-nvmeload"** and **"ipltype-nvmedump"** are valid only when the associated SE version is **2.15.0** with suitable MCL bundle, or a later SE version, and **"ipltype-tape-load"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ccw-load"**, **"ipltype-eckd-ccw-dump"**, **"ipltype-eckd-ld-load"**, and **"ipltype-eckd-ld-dump"** values are only valid on when the associated SE version is **2.16.0** and a sufficient MCL bundle

Name	Type	Rqd/ Opt	Description
`ipl-address`	String (0-5)	Optional	The hexadecimal address of an I/O device that provides access to the control program to be loaded. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. When this property is intended to be used for **ipl-type** of **"ipltype-nvmeload"** or **"ipltype-nvmedump"**, the input value shall be an empty string or a four-digit hexadecimal function ID (FID) of the NVMe device. If it's an FID, it will be right justified and padded with zeros. The values shall be in the range "0000" to "FFFF". For any other **ipl-type**, the input value shall be an empty string or a five-digit hexadecimal address of the I/O device. If it's an address, it will be right justified and padded with zeros. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
`ipl-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank, and period. Three additional characters, (@, $, #) are also allowed. A non-empty string is leading whitespace trimmed and right padded with blanks to 8 characters
`boot-record-location-cylinder`	String (1-7)	Optional	The boot record location cylinder value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`boot-record-location-head`	String (1)	Optional	The boot record location head in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`boot-record-location-record`	String (0-2)	Optional	The boot record location record in hexadecimal. The record may not be set to "0" or "00". **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**

Name	Type	Rqd/ Opt	Description
`boot-record-location-use-volume-label`	Boolean	Optional	Whether the **boot-record-location-cylinder, boot-record-location-head and boot-record-location-record** should be determined by the volume label. On a create, this value cannot be set to **true** if the **boot-record-location-cylinder**, **boot-record-location-head**, or **boot-record-location-record** is specified in the same request. If this value is **true**, it overrides the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** when a load is completed with this profile. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`worldwide-port-name`	String (1-16)	Optional	Worldwide port name value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or**"ipltype-scsidump"**.
`logical-unit-number`	String (1-16)	Optional	Logical unit number value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or**"ipltype-scsidump"**.
`disk-partition-id`	Integer (0-30)	Optional	The disk partition number (also called the boot program selector) for the activation profile, used for a list-directed OS load or dump. This value is ignored when the **disk-partition-id-automatic** value is **true**. When the **disk-partition-id-automatic** property is set to **true**, the value in this field is overridden and not used to complete the load. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, **"ipltype-eckd-ld-dump"**.
`disk-partition-id-automatic`	Boolean	Optional	Whether the value in **disk-partition-id** (also known as the boot program selector) should be ignored, and instead, the value should be determined automatically. When this field is set to **true**, it will override any value in **disk-partition-id**. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, **"ipltype-eckd-ld-dump"**.
`boot-record-lba`	String (1-16)	Optional	Boot record logical block address for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, or **"ipltype-nvmedump"**.

Name	Type	Rqd/ Opt	Description
os-specific-load-parameters	String (0-256)	Optional	Operating system specific load parameters for the activation profile, used for a list-directed load or list-directed dump. On a Create, value is left justified and right-padded with blanks to 256 characters. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **“ipltype-eckd-ld-load”**, or **“ipltype-eckd-ld-dump”**.
secure-boot	Boolean	Optional	If true, the software signature of what is loaded will be checked against what the distributor signed it with. The load will fail if the signatures do not match. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ccwload"**, **"ipltype-eckd-ccwdump"**, **"ipltype-eckd-ld-load"**, **"ipltype-eckd-ld-dump"**.
initial-processing-weight	Integer (1-999)	Optional	The relative amount of shared general purpose processor resources allocated to the logical partition at activation.
number-shared-general-purpose-processors	Integer	Optional	The number of shared general purpose processors allocated to the logical partition.
number-reserved-shared-general-purpose-processors	Integer	Optional	The number of shared general purpose processors reserved for the logical partition.
number-dedicated-general-purpose-processors	Integer	Optional	The number of general purpose processors allocated to the logical partition.
number-dedicated-reserved-general-purpose-processors	Integer	Optional	The number of general purpose processors reserved for the logical partition.

Name	Type	Rqd/ Opt	Description
`intial-processing-weight-capped`	Boolean	Optional	Whether the initial processing weight for general purpose processors is a limit or a target. **true** Indicates that the initial general purpose processing weight for the logical partition is capped. It represents the logical partition's maximum share of general purpose processor resources, regardless of the availability of excess general purpose processor resources. **false** Indicates that the initial general purpose processor processing weight for the logical partition is not capped. It represents the share of general purpose processor resources guaranteed to a logical partition when all general purpose processor resources are in use. Otherwise, when excess general purpose processor resources are available, the logical partition can use them if necessary.
`minimum-processing-weight`	Integer (0-999)	Optional	The minimum relative amount of shared general purpose processor resources allocated to the logical partition. This value must be less than or equal to the value of **maximum-processing-weight.** **0** There is no minimum value for the processing weight.
`maximum-processing-weight`	Integer (0-999)	Optional	The maximum relative amount of shared general purpose processor resources allocated to the logical partition. This value must be greater than or equal to the value of **minimum-processing-weight**. **0** There is no maximum value for the processing weight.
`absolute-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the general purpose processor.
`number-shared-ifl-processors`	Integer	Optional	The number of shared Integrated Facility for Linux (IFL) processors allocated to the logical partition.
`number-reserved-shared-ifl-processors`	Integer	Optional	The number of shared Integrated Facility for Linux (IFL) processors reserved for the logical partition.
`number-dedicated-ifl-processors`	Integer	Optional	The number of Integrated Facility for Linux (IFL) processors allocated to the logical partition.

Name	Type	Rqd/ Opt	Description
`number-reserved-dedicated-ifl-processors`	Integer	Optional	The number of Integrated Facility for Linux (IFL) processors reserved for the logical partition.
`initial-ifl-processing-weight`	Integer (1-999)	Optional	The relative amount of shared Integrated Facility for Linux (IFL) processor resources allocated to the logical partition at activation.
`initial-ifl-processing-weight-capped`	Boolean	Optional	Whether the initial processing weight for Integrated Facility for Linux (IFL) processors is a limit or a target. **true** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Integrated Facility for Linux (IFL) processor resources, regardless of the availability of excess Integrated Facility for Linux (IFL) processor resources. **false** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.
`minimum-ifl-processing-weight`	Integer (0-999)	Optional	The minimum relative amount of shared Integrated Facility for Linux (IFL)processor resources allocated to the logical partition. This value must be less than or equal to the value of **maximum-ifl-processing-weight**. **0** There is no minimum value for the processing weight.
`maximum-ifl-processing-weight`	Integer (0-999)	Optional	The maximum relative amount of shared Integrated Facility for Linux (IFL)processor resources allocated to the logical partition. This value must be greater than or equal to the value of **minimum-ifl-processing-weight**. **0** There is no maximum value for the processing weight.
`absolute-ifl-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
`number-shared-icf-processors`	Integer	Optional	The number of shared Internal Coupling Facility (ICF) processors allocated to the logical partition.

Name	Type	Rqd/ Opt	Description
`number-reserved-shared-icf-processors`	Integer	Optional	The number of shared Internal Coupling Facility (ICF) processors reserved for the logical partition.
`number-dedicated-icf-processors`	Integer	Optional	The number of Internal Coupling Facility (ICF) processors allocated to the logical partition.
`number-reserved-dedicated-icf-processors`	Integer	Optional	The number of Internal Coupling Facility (ICF) processors reserved for the logical partition.
`initial-internal-cf-processing-weight`	Integer (1-999)	Optional	The relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition at activation.
`initial-internal-cf-processing-weight-capped`	Boolean	Optional	Whether the initial processing weight for Internal Coupling Facility (ICF) processors is a limit or a target. **true** Indicates that the initial Internal Coupling Facility (ICF) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of Internal Coupling Facility (ICF) processor resources, regardless of the availability of excess Internal Coupling Facility (ICF) processor resources. **false** Indicates that the initial Integrated Facility for Linux (IFL) processor processing weight for the logical partition is not capped. It represents the share of Integrated Facility for Linux (IFL) processor resources guaranteed to a logical partition when all Integrated Facility for Linux (IFL) processor resources are in use. Otherwise, when excess Integrated Facility for Linux (IFL) processor resources are available, the logical partition can use them if necessary.
`minimum-internal-cf-processing-weight`	Integer (0-999)	Optional	The minimum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition. This value must be less than or equal to the value of **maximum-internal-cf-processing-weight**. **0** There is no minimum value for the processing weight.
`maximum-internal-cf-processing-weight`	Integer (0-999)	Optional	The maximum relative amount of shared Internal Coupling Facility (ICF) processor resources allocated to the logical partition. This value must be greater than or equal to the value of **minimum-internal-cf-processing-weight**. **0** There is no maximum value for the processing weight.

Name	Type	Rqd/ Opt	Description
`absolute-internal-cf-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.
`number-shared-ziip-processors`	Integer	Optional	The number of shared z Integrated Information Processors (zIIP) processors allocated to the logical partition.
`number-reserved-ziip-processors`	Integer	Optional	The number of shared z Integrated Information Processors (zIIP) processors reserved for the logical partition.
`number-dedicated-ziip-processors`	Integer	Optional	The number of z Integrated Information Processors (zIIP) processors allocated to the logical partition.
`number-reserved-dedicated-ziip-processors`	Integer	Optional	The number of z Integrated Information Processors (zIIP) processors reserved for the logical partition.
`initial-ziip-processing-weight`	Integer (1-999)	Optional	The relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition at activation.
`initial-ziip-processing-weight-capped`	Boolean	Optional	Whether the initial processing weight for z Integrated Information Processors (zIIP) processors is a limit or a target. **true** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is capped. It represents the logical partition's maximum share of z Integrated Information Processors (zIIP) processor resources, regardless of the availability of excess z Integrated Information Processors (zIIP) processor resources. **false** Indicates that the initial z Integrated Information Processors (zIIP) processor processing weight for the logical partition is not capped. It represents the share of z Integrated Information Processors (zIIP) processor resources guaranteed to a logical partition when all z Integrated Information Processors (zIIP) processor resources are in use. Otherwise, when excess z Integrated Information Processors (zIIP) processor resources are available, the logical partition can use them if necessary.

Name	Type	Rqd/ Opt	Description
`minimum-ziip-processing-weight`	Integer (0-999)	Optional	The minimum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. This value must be less than or equal to the value of **maximum-ziip-processing-weight** **0** There is no minimum value for the processing weight.
`maximum-ziip-processing-weight`	Integer (0-999)	Optional	The maximum relative amount of shared z Integrated Information Processors (zIIP) processor resources allocated to the logical partition. This value must be greater than or equal to the value of **minimum-ziip-processing-weight**. **0** There is no maximum value for the processing weight.
`absolute-ziip-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the Integrated Facility for Linux(IFL) processor.
`workload-manager-enabled`	Boolean	Optional	Whether or not z/OS Workload Manager is allowed to change processing weight related properties. **true** Indicates that z/OS Workload Manager is allowed to change processing weight related properties for this logical partition. **false** Indicates that z/OS Workload Manager is not allowed to change processing weight related properties for this logical partition.
`crypto-controls-domain`	Array of Integer	Optional	The crypto control domain index. **Note:** If included in the request, this array must contain at least one value. Valid values are unique entries from 0 through a maximum set by the CPC.
`crypto-usages-domain`	Array of Integer	Optional	The crypto usage domain index. **Note:** Values are invalid if there are no matching values in the **crypto-controls-domain**. Valid values are unique entries from 0 through a maximum set by the CPC.
`crypto-candidate-list`	Array of Integer	Optional	The crypto candidate list **Note:** Values are unique entries from 0 through a maximum set by the CPC.
`crypto-online-list`	Array of Integer	Optional	The crypto target online mask. **Note:** Values are invalid if there are no matching values in the **crypto-candidate-list**. Valid values are unique entries from 0 through a maximum set by the CPC.

Name	Type	Rqd/ Opt	Description
`time-offset-days`	Integer (0-999)	Optional	The number of days the logical partition's clock is to be offset from the External Time Source's time of day.
`time-offset-hours`	Integer (0-23)	Optional	The number of hours the logical partition's clock is to be offset from the External Time Source's time of day.
`time-offset-minutes`	Integer Enum	Optional	The number of minutes the logical partition's clock is to be offset from the External Time Source's time of day. Allowable values are **0**, **15**, **30** or **45**.
`time-offset-increase-decrease`	String Enum	Optional	One of: • **"increase"** - Set the logical partition's clock ahead of the External Time Source's time of day. • **"decrease"** - Set the logical partition's clock back from the External Time Source's time of day.
`ssc-host-name`	String (1-64)	Optional	The Secure Service Container host name. Valid characters are: a-z, A-Z,0-9, period(.), minus(-) and colon(:). This is only used when operating-mode is **"ssc"**.
`ssc-primary-userid`	String (1-32)	Optional	The Secure Service Container master user ID. Valid characters are: a-z,A-Z, 0-9, period(.), minus(-) and underscore(_). This is only used when operating-mode is **"ssc"**.
`ssc-primary-pw`	String (8-256)	Optional	The Secure Service Container master password. Valid characters are: a-z,A-Z, 0-9, and !@#$%^&*()_+{}|<>?-=. This is only used when operating-mode is **"ssc"**.
`ssc-network-info`	Array of ssc-network objects	Optional	The set of networks available to the Secure Service Container. A minimum of 1 network and a maximum of 100 networks are permitted. This is only used when operating-mode is **"ssc"**.
`ssc-gateway-info`	ip-info object	Optional	The IPv4 default gateway IP address information for the Secure Service Container. A null object indicates no IPv4 default gateway IP address is specified. This is only used when operating-mode is **"ssc"**.
`ssc-gateway-ipv6-info`	ip-info object	Optional	The IPv6 default gateway IP address information for the Secure Service Container. A null object indicates no IPv6 default gateway IP address is specified. This is only used when operating-mode is **"ssc"**.
`ssc-dns-info`	Array of ip-info object	Optional	The DNS IP address information for the Secure Service Container. A minimum of 0 entries and a maximum of 2 entries are permitted. This is only used when operating-mode is **"ssc"**.

Name	Type	Rqd/ Opt	Description
`ssc-boot-selection`	String Enum	Optional	Indicates whether to run the Secure Service Container appliance installer or the Secure Service Container appliance itself. One of: • **"installer"** - Boot the Secure Service Container appliance installer to install the Secure Service Container appliance and then start it. • **"appliance"** - Reload the most recently installed Secure Service Container appliance and resume its execution from where it was when the image was deactivated.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The element URI of the newly created Image activation profile.

Description

The `Create Image Activation Profile` operation creates a new image profile on the Support Element. This new profile will be created with the name indicated in the **name** field and will be created using the values from the DEFAULT Image profile, unless the **copy-name** field is provided in the request, in which case the profile will be created with the values from the **copy-name** identified profile as initial values. The **copy-name** field, if provided, must contain a valid existing profile name of the same type. Users of this operation can override specific properties by specifying them in the request body.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. On successful execution, the Image Activation Profile is created, and a status code 201 (Created) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned, and the response body is provided as described in “Response body contents” on page 1336, and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The **name** value for this profile type already exists on the CPC with **object-id** *{cpc-id}*.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The object designated by the request URI does not support the requested operation.
	260	The activation profile name in the copy-name field does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	399	The operation cannot be performed as it would exceed the maximum number of Image Activation profiles allowed on the CPC. Retry the operation after an Image Activation profile is deleted from the CPC.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/image-activation-profiles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
Request Body:
{
     "profile-name":"TESTIMG1",
     "copy-name":"DEFAULT"
}
```

Figure 686. Create Image Activation Profile: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/image-activation-profiles
cache-control: no-cache
date: Tue, 16 May 2023 01:35:07 GMT
content-type: application/json;charset=UTF-8
content-length: 99
Response Body:
{
    "element-uri":"/api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/image-activation-profiles/
TESTIMG1"
}
```

Figure 687. Create Image Activation Profile: Response

Delete Image Activation Profile

The `Delete Image Activation Profile` operation deletes a Image activation profile, by profile name designated by *{image-activation-profile-name}* from the Support Element. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
DELETE /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}
```

URI variables:

Table 536.

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{image-activation-profile-name}	Image Activation Profile name.

Description

The `Delete Image Activation Profile` operation deletes a Image Activation profile from the SE. This profile will be deleted based on the *{image-activation-profile-name}*.

The URI path must designate an existing Image Activation Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{image-activation-profile-name}*) does not designate an existing Image Activation Profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/load-activation-profiles/TESTIMG1
x-api-session: 5hirq4lowb9dweimnmoedk5u1qk2ltlfo9f56xrs9kux6w6gzv
```

Figure 688. Delete Image Activation Profile: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 16 May 2023 00:33:27 GMT

<No response body>
```

Figure 689. Delete Image Activation Profile: Response

Inventory service data

Information about image activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Image Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular image activation profile is included only if the API user has access permission to that object as described in the `Get Image Activation Profile Properties` operation.

For each Image Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Image Activation Profile Properties" on page 1310. That is, the data provided is the same as would be provided if a `Get Image Activation Profile Properties` operation were requested targeting this object.

Load activation profile

A Load activation profile is used to load a previously activated logical partition with a control program or operating system.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This element includes the following properties.

Table 537. Load activation profile: properties

Name	Qualifier	Type	Description
element-uri	—	String/ URI	The canonical URI path of the Load Activation Profile object, of the form `/api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}` where *{load-activation-profile-name}* is the value of the name property (Load Activation Profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object
class	—	String	The class of a Load Activation Profile object is **"load-activation-profile"**.

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
name	(pc)	String (1-16)	The activation profile name, which uniquely identifies this profile within the set of activation profiles for the CPC object designated by *{cpc-id}*.
description	(w)	String (1-50)	The load profile description
ipl-address	(w)(pc)	String (0-5)	The hexadecimal address of an I/O device that provides access to the control program to be loaded. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Get: If this property contains an address, the address is either four- or five-digit (in hex) depending on its actual value and the value of **ipl-type**. Update: When this property is intended to be used for **ipl-type** of **"ipltype-nvmeload"** or **"ipltype-nvmedump"**, the input value shall be an empty string or a four-digit hexadecimal function ID (FID) of the NVMe device. If it's a FID, it will be right-justified and padded with zeros. The values shall be in the range "0000" to "FFFF". For any other **ipl-type**, the input value shall be an empty string or a five-digit hexadecimal address of the I/O device. If it's an address, it will be right-justified and padded with zeros. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF". [Updated by feature **secure-boot-with-certificates**]
ipl-parameter[1]	(w)(pc)	String (0-8)	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank, and period. Three additional characters, (@, $, #) are also allowed when the **se-version** property of the associated CPC is **"2.14.0"** or later. On an Update, a non-empty string is left justified and right padded with blanks to 8 characters.

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
ipl-type[2]	(w)(pc)	String Enum	One of: • **"ipltype-scsi"** - This load activation profile is used to perform a SCSI list-directed OS load. • **"ipltype-scsidump"** - This load activation profile is used to perform a SCSI list-directed OS dump. • **"ipltype-nvmeload"** – This load activation profile is used to perform an NVMe list-directed OS load. • **"ipltype-nvmedump"** – This load activation profile is used to perform an NVMe list-directed OS dump. • **"ipltype-tape-load"** - This load activation profile is used to perform a tape Channel Command Word (CCW) OS load. • **"ipltype-tape-dump"** - This load activation profile is used to perform a tape Channel Command Word (CCW) OS dump. • **"ipltype-eckd-ccw-load"** - This load activation profile is used to perform an ECKD Channel Command Word OS load. • **"ipltype-eckd-ccw-dump"** - This load activation profile is used to perform an ECKD Channel Command Word OS dump. • **"ipltype-eckd-ld-load"** - This load activation profile is used to perform an ECKD list-directed OS load. • **"ipltype-eckd-ld-dump"** - This load activation profile is used to perform an ECKD list-directed OS dump. • **"ipltype-standard"** - This load activation profile is used to perform a Channel Command Word (CCW) standard load. This is present for compatibility, and the more specific **"ipltype-tape-load" "ipltype-tape-dump" "ipltype-eckd-ccw-load"** or **"ipltype-eckd-ccw-dump"** is preferred instead on Support Elements with an SE version of 2.16.0 with a sufficient MCL bundle. In that case, if a value of **ipltype-standard** is provided in an update, the value will be interpreted to one of the new values. **"ipltype-eckd-ccw-dump"** will be set if the user specifies the **store-status-indicator** to **true**, otherwise **"ipltype-eckd-ccw-load"** will be set. Any operation that previously worked with **"ipltype-standard"** will continue to work with this translation. For an Update operation, **"ipltype-nvmeload"** and **"ipltype-nvmedump"** are valid only when the associated SE version is 2.15.0 with suitable MCL bundle, or a later SE version, and **"ipltype-tape-load"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ccw-load"**, **"ipltype-eckd-ccw-dump"**, **"ipltype-eckd-ld-load"**, **"ipltype-eckd-ld-dump"** values are only valid on when the associated SE version is 2.16.0 with a sufficient MCL bundle.. [Updated by feature **secure-boot-with-certificates**] [Updated by feature **secure-boot-with-certificates**]

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
secure-boot[3]	(w)(pc)	Boolean	If **true**, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the logical partition. The load will fail if the signatures do not match. This property is only applied to **ipl-type** of **ipltype-scsi**, **ipltype-scsidump**, **ipltype-nvmeload**, **ipltype-nvmedump**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
worldwide-port-name[1]	(w)	String (1-16)	Worldwide port name value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.
disk-partition-id	(w)	Integer (0-30)	Disk partition number (also called the boot program selector) for the activation profile, used for list-directed OS load or dump. When the **disk-partition-id-automatic** property is set to **true**, the value in this field is overridden and not used in the load. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
disk-partition-id-automatic	(w)	Boolean	Whether the value for **disk-partition-id** (also known as the boot program selector) used for a load should be determined automatically. When this field is set to **true**, it will override any value in **disk-partition-id** when a load is completed using this profile. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. This field is only permitted when the SE version of the owning CPC is 2.16.0 with the suitable MCL bundle or later. [Added by feature **secure-boot-with-certificates**]
logical-unit-number[1]	(w)	String (1-16)	Logical unit number value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** Note: This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.
boot-record-lba[1]	(w)	String (1-16)	Boot record logical block address for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, or **"ipltype-nvmedump"**.

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
os-specific-load-parameters	(w)	String (0-256)	Operating system-specific load parameters for the activation profile, used for list-directed OS load or dump. On an Update, value is left justified and right-padded with blanks to 256 characters. **Note:** This property is only applied to **ipl-type** of "**ipltype-scsi**", **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**. [Updated by feature **secure-boot-with-certificates**]
clear-indicator[2]	(w)(pc)	Boolean	Whether memory should be cleared before performing the Load (**true**) or not cleared (**false**). This property is only applied to **ipltype** of **"ipltype-standard"**, **"ipltype-scsi"**, **"ipltype-nvmeload"**, **"ipltype-eckd-ld-load"**, or **"ipltype-tape-load"**. When **ipl-type** is **"ipltype-scsi"** and the **se-version** property of the associated CPC is **"2.14.0"** or earlier, this property cannot be set to **false**. When **ipl-type** is **"ipltype-scsidump"**, this property has no effect, and it cannot be set to **false** due to design constraints. When **ipl-type** is **"ipltype-nvmedump"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ccw-dump"**, or **"ipltype-eckd-ld-dump"**, this property has no effect, and it cannot be set to **true** . [Updated by feature **secure-boot-with-certificates**]
store-status-indicator[2]	(w)	Boolean	When **ipl-type** is **"ipltype-standard"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ld-dump"**, or **"ipltype-eckd-ccw-dump"**, this property indicates whether the store status function should be invoked before performing the dump (**true**) or not (**false**). The store status function stores the current values of the processing unit timer, the clock comparator, the program status word, and the contents of the processor registers in their assigned absolute storage locations. This property cannot be set to **true** when **clear-indicator** is **true**. When **ipl-type** is **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **ipltype-tape-load"**, **"ipltype-eckd-ccw-load"** or **"ipltype-eckd-ld-load**, this property has no effect, and if set, may only be set to **false**. [Updated by feature **secure-boot-with-certificates**]
boot-record-location-cylinder	(w)	String (1-7)	The boot record location cylinder value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC is 2.16.0 with the suitable MCL bundle or later. [Added by feature **secure-boot-with-certificates**]

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
boot-record-location-head	(w)	String (1)	The boot record location head value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC is 2.16.0 with the suitable MCL bundle or later. [Added by feature **secure-boot-with-certificates**]
boot-record-location-record	(w)	String (0-2)	The boot record location record in hexadecimal. The record may not be set to "0" or "00". **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC is 2.16.0 with the suitable MCL bundle or later. [Added by feature **secure-boot-with-certificates**]
boot-record-location-use-volume-label	(w)	Boolean Invalid otherwise	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** should be determined by the volume label. If this value is **true**, it overrides the **boot-record-location-cylinder, boot-record-location-head**, and **boot-record-location-record** when a load is completed with this profile. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"** and only when the SE version of the owning CPC is 2.16.0 with the suitable MCL bundle or later. [Added by feature **secure-boot-with-certificates**]
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Table 537. Load activation profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**

1. An Update request accepts any mixture of [a-f, A-F, 0-9], however the original string value is not saved and a subsequent Get request may not return the exact same set of lower/upper case letters.
2. If the **clear-indicator** or **store-status-indicator** properties are not included in the request body of an `Update Load Activation Profile Properties` operation, certain conditions will cause their values to be set by the system as follows:
   - When **ipl-type** is **"ipltype-standard"** and **clear-indicator** is **true**, the **store-status-indicator** will be set to **false.**
   - When **ipl-type** is **"ipltype-scsi"**, the **store-status-indicator** will be set to **false**. When the **se-version** property of the associated CPC is **"2.14.0"** or earlier, the **clear-indicator** will be set to **true**.
   - When **ipl-type** is **"ipltype-scsidump"**, the **clear-indicator** will be set to **true** and the **store-status-indicator** will be set to **true**.
   - When **ipl-type** is **"ipltype-nvmeload"**, the **store-status-indicator** will be set to **false**.
   - When **ipl-type** is **"ipltype-nvmedump"**, the **clear-indicator** will be set to **false** and the **store-status-indicator** will be set to **false**.
   - When **ipl-type** is being changed from **"ipltype-scsidump"** to **"ipltype-standard"**, the **store-status-indicator** will be set to **false**.

   It is recommended that an `Update Load Activation Profile Properties` operation that includes **ipl-type**, **clear-indicator** or **store-status-indicator** should include all three of those properties to ensure that the desired values are in effect.
3. On a Get request, this property is returned only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version. On an Update request, this property can be included in the request body only when the associated SE version is 2.15.0 with the suitable MCL bundle, or a later SE version.

List Load Activation Profiles

The `List Load Activation Profiles` operation lists the Load Activation Profiles for the associated CPC object. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/load-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
name	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`load-activation-profiles`	Array of load-actprof-info objects	Array of nested objects (described in the next table).

Each load-actprof-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/URI	Canonical URI path of the Load Activation Profile object.
`name`	String	The name of the Load Activation Profile.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Load Activation Profiles for the associated CPC object.

If the name query parameter is specified, the returned list is limited to those Load Activation Profiles that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1346.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to at least one logical partition of that CPC.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1346.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the collection of the list of activation profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 690. List Load Activation Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json;charset=UTF-8
content-length: 363
{
   "load-activation-profiles": [
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-
           profiles/DEFAULTLOAD",
         "name": "DEFAULTLOAD"
      },
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-
           profiles/MODIFYL",
         "name": "MODIFYL"
      }
   ]
}
```

Figure 691. List Load Activation Profiles: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI through the `List CPC Objects` operation. Instead, such a user may use either the `List Permitted Logical Partitions` operation or the `Get Inventory` operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Get Load Activation Profile Properties

The `Get Load Activation Profile Properties` operation retrieves the properties of a single Load Activation Profile designated by *{load-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}
```

URI variables

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{load-activation-profile-name}	Load Activation Profile name.

Query parameters:

Name	Type	Rqd/Opt	Description
`cached-acceptable`	Boolean	Optional	Indicates whether cached values are acceptable for the returned properties. Valid values are **true** and **false**. The default is **false**.
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the object's data model.

Response body contents

On successful completion, the response body provides the current values of the properties for the Load Activation Profile as defined in the "Data model" on page 1340.

Description

The URI path must designate an existing Load Activation Profile and the API user must have object-access permission to the associated CPC object or at least one of its logical partitions. If either of these conditions is not met, status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

Some of this object's property values are periodically fetched from the Support Element and cached for quick access by the APIs. Due to the nature of this caching support, the cached value of a property may differ from the actual value at any point in time. While the cache is kept reasonably current, there are no guarantees about the latency of the cache, nor is there any latency or other cache information available to the API user. If the **cached-acceptable** query parameter is specified as **true** and a property's value is currently present in the cache, the value from the cache is returned; otherwise, the current, non-cached value is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for Load Activation Profiles.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to at least one logical partition of that CPC.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1349.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the CPC or any of its logical partitions.
	260	The activation profile name in the URI (*{load-activation-profile-name}*) does not designate an existing activation profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles/DEFAULTLOAD HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 692. Get Load Activation Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json;charset=UTF-8
content-length: 812
{
   "boot-record-lba": "abcdef0123456789",
   "class": "load-activation-profile",
   "clear-indicator": true,
   "description": "This is the default Load profile.",
   "disk-partition-id": 0,
   "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/load-activation-profiles/
     DEFAULTLOAD",
   "ipl-address": "00D00",
   "ipl-parameter": "        ",
   "ipl-type": "ipltype-scsi",
   "logical-unit-number": "0",
   "name": "DEFAULTLOAD",
   "os-specific-load-parameters": " ",
   "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340",
   "secure-boot": false,
   "store-status-indicator": false,
   "worldwide-port-name": "0"
}
```

Figure 693. Get Load Activation Profile Properties: Response

Usage Note

An API user without object-access permission to the CPC cannot obtain the CPC's URI through the `List CPC Objects` operation. Instead, such a user may use either the `List Permitted Logical Partitions` operation or the `Get Inventory` operation to obtain the properties of a Logical Partition object to which the user has object-access permission. The parent CPC's URI is included in the response body of those operations.

Update Load Activation Profile Properties

The `Update Load Activation Profile Properties` operation updates one or more writable properties of the Load Activation Profile designated by *{load-activation-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}
```

URI variables

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{load-activation-profile-name}	Load Activation Profile name.

Request body contents

The request body is expected to contain one or more field names representing writable Load Activation Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Load Activation Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

To allow the API user to prepare the profile for future use, the input value for a field is not validated against the values of other fields in the profile, unless specified otherwise. This implies that, when the input value of a field is not applicable or is conflicting with the value of another field, the operation still allows the update, and the new value will be saved in the profile. However, the conflicting values among the fields in the profile may cause the Activation or Load operation to fail. It is up to the user to validate the content of the profile before it is actually being used.

On successful execution, the value of each corresponding property of the Load Activation Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Customize/Delete Activation Profiles** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{load-activation-profile-name}*) does not designate an existing activation profile.

HTTP error status code	Reason code	Description
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Create Load Activation Profile

The `Create Load Activation Profile` operation creates a new Load activation profile. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/load-activation-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

Name	Type	Rqd/ Opt	Description
`name`	String (1-16)	Required	The Load activation profile name, which uniquely identifies this profile within the set of Load activation profiles for the CPC object designated by *{cpc-id}*.
`copy-name`	String (1-16)	Optional	The name of an existing Load activation profile on the CPC object designated by *{cpc-id}*. If this field is provided, the name must be a valid name of an existing profile, which will then be loaded as the initial values for the Create operation. If this field is not sent in the request, the new load profile will use the fields from the DEFAULTLOAD load profile as the initial values for the created profile.
`description`	String (0-50)	Optional	The load profile description of the profile to be created.

Name	Type	Rqd/ Opt	Description
`clear-indicator`	Boolean	Optional	Whether memory should be cleared before performing the Load (**true**) or not cleared (**false**). This property is only applied to **ipltype** of **"ipltype-standard"**, **"ipltype-scsi"**, **"ipltype-nvmeload"**, **"ipltype-eckd-ld-load"**, or **"ipltype-tape-load"**. When **ipl-type** is **"ipltype-scsi-dump"**, this property has no effect, and it cannot be set to **false** due to design constraints. When **ipl-type** is **"ipltype-nvmedump"**, **"ipltype-tape-dump"**, **"ipltype-eckd-ccw-dump"**, or **"ipltype-eckd-ld-dump"**, this property has no effect, and it cannot be set to **true**.
`load-timeout`	Integer (60-600)	Optional	Amount of time, in seconds, to wait for the Load to complete.
`store-status-indicator`	Boolean	Optional	When **ipl-type** is **"ipltype-standard"**, **"ipltype-eckd-ld-dump"**, or **"ipltype-tape-dump"** or, **"ipltype-eckd-ccw-dump"**, this property indicates whether the store status function should be invoked before performing the Load (**true**) or not (**false**). The store status function stores the current values of the processing unit timer, the clock comparator, the program status word, and the contents of the processor registers in their assigned absolute storage locations. This property cannot be set to true when **clear-indicator** is **true**. When **ipl-type** is **"ipltype-scsi"**, **"ipltype-nvmeload"**, or **"ipltype-nvmedump"**, **"ipltype-tape-load"**, **"ipltype-eckd-ccw-load"**, or **"ipltype-eckd-ld-load"**, this property has no effect, and if set, may only be set to **false**.

Name	Type	Rqd/ Opt	Description
`ipl-type`	String Enum	Optional	One of: • **"ipltype-scsi"** - This load activation profile is used to perform a SCSI list-directed OS load. • **"ipltype-scsidump"** - This load activation profile is used to perform a SCSI list-directed OS dump. • **"ipltype-nvmeload"** – This load activation profile is used to perform an NVMe list-directed OS load. • **"ipltype-nvmedump"** – This load activation profile is used to perform an NVMe list-directed OS dump. • **"ipltype-tape-load"** - This load activation profile is used to perform a tape Channel Command Word (CCW) OS load • **"ipltype-tape-dump"** - This load activation profile is used to perform a tape Channel Command Word (CCW) OS dump • **"ipltype-eckd-ccw-load"** - This load activation profile is used to perform an ECKD Channel Command Word OS load. • **"ipltype-eckd-ccw-dump"** - This load activation profile is used to perform an ECKD Channel Command Word OS dump • **"ipltype-eckd-ld-load"** - This load activation profile is used to perform an ECKD list-directed OS load • **"ipltype-eckd-ld-dump"** - This load activation profile is used to perform an ECKD list-directed OS dump • **"ipltype-standard"** - This load activation profile is used to perform a Channel Command Word (CCW) standard load. This is present for compatibility, and the more specific "**ipltype-tape-load" "ipltype-tape-dump" "ipltype-eckd-ccw-load"** or **"ipltype-eckd-ccw-dump"** is preferred. In that case, if a value of **ipltype-standard** is provided in an update, the value will be interpreted to one of the new values. **ipltype-eckd-ccw-dump** will be set if the user specifies the **store-status-indicator** to true, otherwise **ipltype-eckd-ccw-load** will be set. Any operation that previously worked with **"ipltype-standard"** will continue to work with this translation

Name	Type	Rqd/ Opt	Description
`ipl-address`	String (0-5)	Optional	The hexadecimal address of an I/O device that provides access to the control program to be loaded. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. When this property is intended to be used for ipl-type of **"ipltype-nvmeload"** or **"ipltype-nvmedump"**, the input value shall be an empty string or a four-digit hexadecimal function ID (FID) of the NVMe device. If it's an FID, it will be right justified and padded with zeros. The values shall be in the range "0000" to "FFFF". For any other **ipl-type**, the input value shall be an empty string or a five-digit hexadecimal address of the I/O device. If it's an address, it will be right justified and padded with zeros. Valid values are in the range "00000" to "*n*FFFF" where *n* is the number of subchannel sets provided by the CPC minus 1. So, for example, on a CPC that provides 3 subchannel sets, the valid range is "00000" to "2FFFF".
`ipl-parameter`	String (0-8)	Optional	Some control programs support the use of this property to provide additional control over the outcome of a Load operation. Refer to the configuration documentation for the control program to be loaded to see if this parameter is supported and if so, what values and format is supported. An empty string indicates that the value for this property is to be retrieved from the IOCDS used during a subsequent Load operation. Valid characters are 0-9, A-Z, blank, and period. Three additional characters, (@, $, #) are also allowed. A non-empty string is leading whitespace trimmed and right padded with blanks to 8 characters
`boot-record-location-cylinder`	String (1-7)	Optional	The boot record location cylinder value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`boot-record-location-head`	String (1)	Optional	The boot record location head value in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`boot-record-location-record`	String (0-2)	Optional	The boot record location record in hexadecimal. The record may not be set to “0” or “00”. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.

Name	Type	Rqd/ Opt	Description
`boot-record-location-use-volume-label`	Boolean	Optional	Whether the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** should be determined by the volume label. If this value is **true**, it overrides the **boot-record-location-cylinder**, **boot-record-location-head**, and **boot-record-location-record** when a load is completed with this profile. **Note:** This property is only applied to **ipl-type** of **"ipltype-eckd-ld-load"** and **"ipltype-eckd-ld-dump"**.
`worldwide-port-name`	String (1-16)	Optional	Worldwide port name value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. Note: This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.
`logical-unit-number`	String (1-16)	Optional	Logical unit number value for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.
`disk-partition-id`	Integer (0-30)	Optional	Disk partition number (also called the boot program selector) for the activation profile, used for a list-directed load or dump. When the **disk-partition-id-automatic** value is set to **true**, the value in this field is overridden and not used in the load. Note: This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**.
`disk-partition-id-automatic`	Boolean	Optional	Whether the value in **disk-partition-id** (also known as the boot program selector) used for a load should be determined automatically. When this field is set to **true,** it will override any value in **disk-partition-id** when a load is completed using this profile. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**.
`boot-record-lba`	String (1-16)	Optional	Boot record logical block address for the activation profile, used for SCSI load and SCSI dump, in hexadecimal. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"** or **"ipltype-scsidump"**.

Name	Type	Rqd/ Opt	Description
`os-specific-load-parameters`	String (0-256)	Optional	Operating system-specific load parameters for the activation profile, used for a list-directed load or dump. On Create, value is left justified and right-padded with blanks to 256 characters. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**.
`secure-boot`	Boolean	Optional	If **true**, the software signature of the operating system or dump program will be verified using the certificate(s) assigned to the logical partition. The load will fail if the signatures do not match. **Note:** This property is only applied to **ipl-type** of **"ipltype-scsi"**, **"ipltype-scsidump"**, **"ipltype-nvmeload"**, **"ipltype-nvmedump"**, **"ipltype-eckd-ld-load"**, or **"ipltype-eckd-ld-dump"**.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The element URI of the newly created Load activation profile.

Description

The **Create Load Activation Profile** operation creates a new load profile on the SE. This new profile will be created with the name indicated in the **name** field and will be created using the values from the DEFAULTLOAD Load profile as the initial field values, unless the **copy-name** field is provided in the request, in which case the profile will be created with the values from the **copy-name** identified profile as initial values. The **copy-name** field, if provided, must contain a valid existing profile name of the same type. Users of this operation can override specific properties by specifying them in the request body.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. On successful execution, the Load Activation Profile is created, and a status code 201 (Created) is returned

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned, and the response body is provided as described in "Response body contents" on page 1358, and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The **name** value for this profile type already exists on the CPC with **object-id** *{cpc-id}*.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The object designated by the request URI does not support the requested operation.
	260	The activation profile name in the copy-name field does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	399	The operation cannot be performed as it would exceed the maximum number of Load Activation profiles allowed on the CPC. Retry the operation after a Load Activation profile is deleted from the CPC.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/load-activation-profiles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
Request Body:
{
     "profile-name":"TESTLOA1",
     "copy-name":"DEFAULT"
}
```

Figure 694. Create Load Activation Profile: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/users/e9e8d20a-4a7a-11e4-91ee-1c6f65065a91
cache-control: no-cache
date: Tue, 16 May 2023 01:35:07 GMT
content-type: application/json;charset=UTF-8
content-length: 98
Response Body:
{
    "element-uri":"/api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/load-activation-profiles/
TESTLOA1"
}
```

Figure 695. Create Load Activation Profile: Response

Delete Load Activation Profile

The `Delete Load Activation Profile` operation deletes a Load activation profile, by profile name designated by *{load-activation-profile-name}* from the Support Element. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

`DELETE /api/cpcs/`*`{cpc-id}`*`/load-activation-profiles/`*`{load-activation-profile-name}`*

URI variables:

Table 538.

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{load-activation-profile-name}	Load Activation Profile name.

Description

The `Delete Load Activation Profile` operation deletes a Load Activation profile from the SE. This profile will be deleted based on the *{load-activation-profile-name}*.

The URI path must designate an existing Load Activation Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{load-activation-profile-name}*) does not designate an existing Load Activation Profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/load-activation-profiles/TESTLOA1
x-api-session: 5hirq4lowb9dweimnmoedk5u1qk2ltlfo9f56xrs9kux6w6gzv
```

Figure 696. Delete Load Activation Profile: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 16 May 2023 00:33:27 GMT

<No response body>
```

Figure 697. Delete Load Activation Profile: Response

Inventory service data

Information about load activation profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Load Activation Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular load activation profile is included only if the API user has access permission to that object as described in the `Get Load Activation Profile Properties` operation.

For each Load Activation Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Load Activation Profile Properties" on page 1349. That is, the data provided is the same as would be provided if a `Get Load Activation Profile Properties` operation were requested targeting this object.

Group profile

A Group profile is used to define the group capacity value for all logical partitions belonging to that group.

A logical partition becomes a member of a group profile by placing the group profile's URI in the image activation profile used to activate the logical partition.

For information on customizing activation profiles, Support Element (Version 2.12.1 and newer) information can be found on console help system. For information from earlier versions of the Support Element, see the *Support Element Operations Guide*.

The Group Profile object's data model includes effective properties, denoted by the (e) qualifier. Those properties are applicable when at least one of the member logical partitions is currently activated; the **effective-properties-apply** property has a value of **true** when that is the case. Changing the value of **effective-logical-partition-names** updates the names of the **active** logical partition objects that are a member of the group. Changing the value of other effective properties updates the corresponding characteristic of currently active logical partitions in the group, and the updated value applies to subsequently activated logical partitions in the group. Once the last active logical partition in the group has been deactivated, all changes to effective properties will be discarded and the effective properties return to their base value.

Objects of this class are not provided when the CPC is enabled for DPM.

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This element includes the following properties.

Table 539. Group profile: properties			
Name	**Qualifier**	**Type**	**Description**
element-uri	—	String/ URI	The canonical URI path of the group profile object, of the form `/api/cpcs/{cpc-id}/group-profiles/{group-profile-name}` where *{group-profile-name}* is the value of the name property (group profile name).
parent	—	String/ URI	The canonical URI path of the associated CPC object
class	—	String	The class of a Group Profile object is **"group-profile"**.
name	—	String (1-16)	The group profile name, which uniquely identifies this profile within the set of group profiles for the CPC object designated by *{cpc-id}*
description	(w)	String (1-50)	The group profile description
capacity	(w)	Integer	The upper bound, in MSUs, beyond which the rolling 4-hour average CPU utilization cannot exceed for the group. A value of 0 indicates the setting is unused.
effective-capacity	(e)(w)	Integer	The effective value of the **capacity** property.
absolute-icf-capping[1]	(w)	absolute-capping-object	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.
effective-absolute-icf-capping[1]	(e)(w)	absolute-capping-object	The effective value of the **absolute-icf-capping** property.
absolute-ifl-capping[1]	(w)	absolute-capping-object	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
effective-absolute-ifl-capping[1]	(e)(w)	absolute-capping-object	The effective value of the **absolute-ifl-capping** property.
absolute-general-purpose-capping[1]	(w)	absolute-capping-object	The amount of absolute capping applied to the general purpose processor.
effective-absolute-general-purpose-capping[1]	(e)(w)	absolute-capping-object	The effective value of the **absolute-general-purpose-capping** property.
absolute-ziip-capping[1]	(w)	absolute-capping-object	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor.
effective-absolute-ziip-capping[1]	(e)(w)	absolute-capping-object	The effective value of the **absolute-ziip-capping** property.

Table 539. Group profile: properties (continued)			
Name	**Qualifier**	**Type**	**Description**
effective-properties-apply	—	Boolean	Indicates whether the object is currently in a state in which effective properties are applicable (**true**). Otherwise, the value is **false**.
logical-partition-names	(w)	Array of String	The names of the Logical Partition objects that are a member of the group.
effective-logical-partition-names	(e)(w)	Array of String	The effective value of the **logical-partition-names** property. **Note:** The Logical Partition objects in this list shall be active.
target-name	—	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.
Notes: 1. This property is only provided when the associated SE is at version 2.13.1 or later.			

List Group Profiles

The `List Group Profiles` operation lists the Group Profiles for the associated CPC object. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/group-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Query parameters:

Name	Type	Rqd/ Opt	Description
`name`	String	Optional	A regular expression used to limit returned objects to those that have a matching name property. If matches are found, the response will be an array with all objects that match. If no match is found, the response will be an empty array.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`group-profiles`	Array of group-actprof-info objects	Array of nested objects (described in the next table).

Each group-actprof-info object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path of the Group Profile object.
`name`	String	The name of the Group Profile.
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the Group Profiles for the associated CPC object.

If the name query parameter is specified, the returned list is limited to those Group Profiles that have a name property matching the specified filter pattern. If the name parameter is omitted, this filtering is not done.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*, or object-access permission to the same named logical partition of that CPC, for each object to be included in the response.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1364.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	299	A query parameter has an invalid syntax.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

HTTP error status code	Reason code	Description
500 (Server Error)	281	An unexpected error occurred during the collection of the list of group profiles.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 698. List Group Profiles: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:19 GMT
content-type: application/json;charset=UTF-8
content-length: 182
{
   "group-profiles": [
      {
         "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/
           DEFAULT",
         "name": "DEFAULT"
      }
   ]
}
```

Figure 699. List Group Profiles: Response

Get Group Profile Properties

The `Get Group Profile Properties` operation retrieves the properties of a single Group Profile designated by *{group-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}
```

URI variables

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{group-profile-name}	Group Profile name.

Query parameters:

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the object's data model.

Response body contents

On successful completion, the response body provides the current values of the properties for the Group Profile as defined in the "Data model" on page 1362.

Description

The URI path must designate an existing Group Profile and the API user must have object-access permission to the associated CPC object. If either of these conditions is not met, HTTP status code 404 (Not Found) is returned.

If the **properties** query parameter is specified, the response body contains only the requested properties. The presence and value of each requested property is the same as it is when the **properties** query parameter is not specified. That is, it may be omitted or contain a special value, such as null, -1, or an empty string, if a prerequisite condition is not met. If the **properties** parameter is omitted, no such filtering is performed.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Group Profile.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Object-access permission to the CPC object designated by *{cpc-id}*.
- If any of the **effective-*** properties is to be updated, action/task permission for the **Change LPAR Group Controls** task.
- For all other properties, action/task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1367.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The group profile name in the URI (*{group-profile-name}*) does not designate an existing group profile.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
GET /api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/DEFAULT HTTP/1.1
x-api-session: 5obf0hwsfv1sg9kr5f93cph3zt6o5cptb6lcl538wuyebdyzu4
```

Figure 700. Get Group Profile Properties: Request

```
200 OK
server: zSeries management console API web server / 1.0
cache-control: no-cache
date: Fri, 25 Nov 2011 17:16:20 GMT
content-type: application/json;charset=UTF-8
content-length: 162
{"absolute-icf-capping": {"type": "none"},
   "absolute-ifl-capping": {"value": 98.23, "type": "processors"},
   "absolute-general-purpose-capping": {"value": 1.05, "type": "processors"},
   "absolute-ziip-capping": {"type": "none"},
   "capacity": 0,
   "class": "group-profile",
   "description": "This is the default Group profile.",
   "effective-absolute-icf-capping": {"value": 55.86, "type": "processors"},
   "effective-absolute-ifl-capping": {"value": 98.23, "type": "processors"},
   "effective-absolute-general-purpose-capping": {"type": "none"},
   "effective-absolute-ziip-capping": {"type": "none"},
   "effective-capacity": 1,
   "effective-properties-apply": true,
   "element-uri": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340/group-profiles/DEFAULT",
   "name": "DEFAULT",
   "parent": "/api/cpcs/37c6f8a9-8d5e-3e5d-8466-be79e49dd340"
}
```

Figure 701. Get Group Profile Properties: Response

Update Group Profile Properties

The `Update Group Profile Properties` operation updates one or more writable properties of the Group Profile object designated by *{group-profile-name}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}
```

URI variables

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{group-profile-name}	Group Profile name.

Request body contents

The request body is expected to contain one or more field names representing writable Group Profile properties, along with the new values for those fields.

The request body can and should omit fields for properties whose values are not to be changed by this operation. Properties for which no input value is provided remain unchanged by this operation.

Description

The request body object is validated against the data model for the Group Profile to ensure that the request body contains only writable properties and the data types of those properties are as required. If the request body is not valid, HTTP status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered.

On successful execution, the value of each corresponding property of the Group Profile is updated with the value provided by the input field, and HTTP status code 204 (No Content) is returned.

When this operation changes the value of any property for which property-change notifications are due, those notifications are emitted asynchronously to this operation.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object designated by *{cpc-id}*
  - Action/task permission for the **Customize/Delete Activation Profiles** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The group profile name in the URI (*{group-profile-name}*) does not designate an existing group profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	8	The operation cannot be completed because it is attempting to update an effective property but the supplied value is conflicting with the status of one or more logical partitions, or it would result in another group being placed into a state that is inconsistent with its data model or other requirements.
	9	The operation cannot be completed because it is attempting to update an effective property when the object is not in a state in which effective properties are applicable. More specifically, the request body contains one or more fields which correspond to a property marked with the (e) qualifier in the data model, and the object's **effective-properties-apply** property value is **false**.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Create Group Profile

The `Create Group Profile` operation creates a new Group profile. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/group-profiles
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Request body contents

Name	Type	Rqd/ Opt	Description
`profile-name`	String (1-8)	Required	The Group profile name, which uniquely identifies this profile within the set of Group profiles for the CPC object designated by *{cpc-id}*.
`copy-name`	String (1-8)	Optional	The name of an existing Group profile on the CPC object designated by *{cpc-id}*. If this field is provided, the name must be a valid name of an existing profile, which will then be loaded as the initial values for the Create operation. If this field is not sent in the request, the new group profile will use the fields from the DEFAULT group profile as the initial values for the created profile.
`description`	String (1-50)	Optional	The group profile description of the profile to be created.
`capacity`	Integer (0-2147483647)	Optional	The upper bound, in MSUs, beyond which the rolling 4-hour average CPU utilization cannot exceed for the group. A value of 0 indicates the setting is unused.
`absolute-icf-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the Internal Coupling Facility (ICF) processor.
`absolute-ifl-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the Integrated Facility for Linux (IFL) processor.
`absolute-general-purpose-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the general purpose (CP) processor.
`absolute-ziip-capping`	absolute-capping object	Optional	The amount of absolute capping applied to the z Integrated Information Processors (zIIP) processor.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`element-uri`	String/ URI	The element URI of the newly created Group profile.

Description

The `Create Group Profile` operation creates a new group profile on the Support Element. This new profile will be created with the name indicated in the **name** field and will be created using the values from the DEFAULT Load profile, unless the **copy-name** field is provided in the request, in which case the profile will be created with the values from the **copy-name** identified profile as initial values. The **copy-name** field, if provided, must contain a valid existing profile name of the same type. Users of this operation can override specific properties by specifying them in the request body.

If the request body is not valid, status code 400 (Bad Request) is returned with a reason code indicating the validation error encountered. On successful execution, the Group Profile is created, and a status code 201 (Created) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 201 (Created) is returned, and the response body is provided as described in , and the Location response header contains the URI of the newly created object.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	8	The **name** value for this profile type already exists on the CPC with **object-id** *{cpc-id}*.
	300	The provided update values would result in an illegal state. Verify that the values are both internally consistent and consistent with the current state of the profile.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	4	The object designated by the request URI does not support the requested operation.
	260	The activation profile name in the copy-name field does not designate an existing activation profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
	399	The operation cannot be performed as it would exceed the maximum number of Group profiles allowed on the CPC. Retry the operation after a Group profile is deleted from the CPC.

HTTP error status code	Reason code	Description
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Example HTTP interaction

```
POST /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/image-activation-profiles HTTP/1.1
x-api-session: 2t4ixcf8nplr7yersi8i9b953fgxvvqxl8c4r066ge9kcyzr4c
content-type: application/json
```

Figure 702. Create Group Profile: Request

```
201 Created
server: zSeries management console API web server / 2.0
location: /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/group-profiles
cache-control: no-cache
date: Tue, 16 May 2023 01:35:07 GMT
content-type: application/json;charset=UTF-8
content-length: 88
Response Body:
{
    "element-uri":"/api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/group-profiles/TESTGRP1"
}
```

Figure 703. Create Group Profile: Response

Delete Group Profile

The `Delete Group Profile` operation deletes a Group profile, by profile name designated by *{group-profile-name}* from the Support Element. This operation is supported using the BCPii interface. [Added by feature **create-delete-activation-profiles**]

HTTP method and URI

`DELETE /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`

URI variables:

Table 540.

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{group-profile-name}	Group Profile name.

Description

The `Delete Group Profile` operation deletes a Group profile from the SE. This profile will be deleted based on the *{group-profile-name}*.

The URI path must designate an existing Group Profile and the API user must have object-access permission to the CPC. If either of these conditions is not met, status code 404 (Not Found) is returned.

On success, HTTP status code 204 (No Content) is returned.

Authorization requirements

This operation has the following authorization requirements:

For the web services interface:

- Action/Task permission for the **Customize/Delete Activation Profiles** task.

For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned and no response body is provided.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	260	The activation profile name in the URI (*{group-profile-name}*) does not designate an existing Group Profile.
409 (Conflict)	2	The operation was rejected by the Support Element (SE), because the SE is currently performing processing that requires exclusive control of the SE. Retry the operation at a later time.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	281	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
DELETE /api/cpcs/92525d83-50e8-34a2-8f83-73f375ea85ae/group-profiles/TESTGRP1
x-api-session: 5hirq4lowb9dweimnmoedk5u1qk2ltlfo9f56xrs9kux6w6gzv
```

Figure 704. Delete Group Profile: Request

```
204 No Content
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 16 May 2023 00:33:27 GMT

<No response body>
```

Figure 705. Delete Group Profile: Response

Inventory service data

Information about group profiles can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Group Profile objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular group profile is included only if the API user has access permission to that object as described in the `Get Group Profile Properties` operation.

For each Group Profile object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Group Profile Properties" on page 1366. That is, the data provided is the same as would be provided if a `Get Group Profile Properties` operation were requested targeting this object.

Capacity records

A capacity record represents a temporary upgrade that can be applied to a CPC.

These upgrades are provided through the following offerings:

- **On/Off Capacity on Demand (On/Off CoD)** - This offering allows you to temporarily add additional capacity or specialty engines due to seasonal activities, period-end requirements, peaks in workload, or application testing.
- **Capacity Backup (CBU)** - This offering allows you to replace model capacity or specialty engines to a backup server in the event of an unforeseen loss of server capacity because of an emergency.
- **Capacity for Planned Events (CPE)** - This offering allows you to replace model capacity or specialty engines due to a relocation of workload during system migrations or a data center move.

Data model

For definitions of the qualifier abbreviations in the following tables, see "Property characteristics" on page 98.

This element includes the following class-specific properties.

Table 541. Capacity records: class-specific properties		
Name	**Type**	**Description**
element-uri	String/ URI	The canonical URI path of the capacity record object, of the form `/api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}` where *{cpc-id}* is the value of the **object-id** property of the CPC object and *{capacity-record-id}* is the value of the **record-identifier** property of the Capacity Record object.
parent	String/ URI	The canonical URI path for the associated CPC object
class	String	The class of a capacity record object is **"capacity-record"**.
record-identifier	String (1-8)	The identifier for the capacity record.
record-type	String Enum	The type of capacity record. One of: • **"unknown"** - the record does not specify a record-type • **"cbu"** - a Capacity Backup Upgrade record • **"oocod"** - an On/Off Capacity on Demand record • **"planned-event"** - a Capacity for Planned Events record • **"loaner"** - resources loaned to the installation • **"boost"** - System Recovery Boost record • **"z-flexible-capacity "** - IBM zSystems Flexible Capacity Disaster Recovery • **"z-flexible-capacity-oocod"** - IBM zSystems Flexible Capacity On/Off Capacity on Demand record • **"tfp-hw"** - Tailored Fit Pricing Hardware.
activation-status	String Enum	An indication if any of the resources defined for the record are currently activated. One of: • **"unknown"** - the activation status of the record is not known • **"not-activated"** - the record is not currently active • **"real"** - the record is either active or pending activation, through an Add Temporary Capacity operation with a test=false option • **"test"** - the record is either active or pending activation through an Add Temporary Capacity operation with a test=true option • **"can-be-activated"** - the record is available for activation, but not currently active.
activation-date	Timestamp	Defines the time stamp for when additional capacity for the record was activated.
record-expiration-date	Timestamp	Defines the time stamp for when the capacity record will expire.
activation-expiration-date	Timestamp	Defines the time stamp for when the additional capacity activated for the record will expire and no longer be active.
maximum-real-days	Integer	Defines the maximum days that real additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.

Table 541. Capacity records: class-specific properties (continued)		
Name	**Type**	**Description**
maximum-test-days	Integer	Defines the maximum days that test additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-real-days	Integer	Defines the remaining number of days that additional real capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-test-days	Integer	Defines the remaining number of days that additional test capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
maximum-real-hours	Integer	Defines the maximum hours that real additional capacity can be activated for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
maximum-test-hours	Integer	Defines the maximum hours that test additional capacity can be activated for the record. A value of-1 indicates that this limit is not applicable to this type of capacity record.
remaining-real-hours	Integer	Defines the remaining number of hours that additional real capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-test-hours	Integer	Defines the remaining number of hours that additional test capacity can be active for the record. A value of -1 indicates that this limit is not applicable to this type of capacity record.
remaining-number-of-real-activations	Integer	Defines the number of times that real additional capacity can be activated for the record. A value of -1 indicates that activation count is unlimited.
remaining-number-of-test-activations	Integer	Defines the number of times that test additional capacity can be activated for the record. A value of -1 indicates that activation count is unlimited.
processor-info	Array of caprec-proc-info objects	A nested object describing the processor capacities available with this capacity record.
available-targets	Array of caprec-target objects	A nested object describing the set of possible activation and deactivation targets contained within this capacity record. One of these targets is chosen through the software-model request body field on the `Add Temporary Capacity` or `Remove Temporary Capacity` operations.
target-name	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Table 542. caprec-proc-info object		
Name	**Type**	**Description**
type	String Enum	Identifies the type of specialty processor represented. One of: • **"cp"** - central processor • **"aap"** - Application Assist Processor • **"ifl"** - Integrated Facility for Linux processor • **"icf"** - Internal Coupling Facility processor • **"iip"** - z Integrated Information Processors processor • **"sap"** - System Assist Processor
processor-step	Integer	The number of processors steps available
speed-step	Integer	The CP processor speed activation step. A null object is returned for all other processor types.
max-number-processors	Integer	The maximum number of processors available for this processor type. A value of -1 indicates an unlimited number of processors.
remaining-processor-days	Integer	The remaining processor days for this processor type. A value of -1 indicates an unlimited number of days.
remaining-msu-days	Integer	The remaining MSU days for this processor type. A value of -1 indicates an unlimited number of days. A null object is returned for processor types where this field is not meaningful.

Table 543. caprec-target object		
Name	**Type**	**Description**
processor-step	Integer	The CPU processor step. This is the incremental delta CPUs compared to the current activation level. The returned value may be negative.
speed-step	Integer	The CPU processor speed activation step. This is the incremental delta speed steps compared to current activation level. The returned value may be negative.
software-model	String (1-3)	The software model that this target represents
billable-msu-cost	Integer	The overall billable MSU cost for this target
billable-msu-delta	Integer	The change in billable MSU cost by activating this target. The value may be negative.

List Capacity Records

The `List Capacity Records` operation lists the capacity record for a given CPC that are managed by this HMC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-records
```

In this request, the URI variable *{cpc-id}* is the object ID of the target CPC object.

Response body contents

On successful completion, the response body contains a JSON object with the following fields:

Field name	Type	Description
`capacity-record`	Array of objects	Array of nested objects (described in the next table).

Each nested object contains the following fields:

Field name	Type	Description
`element-uri`	String/ URI	Canonical URI path of the Capacity Record object.
`record-identifier`	String	The record identifier of the Capacity Record
`target-name`	String (1-17)	The value that must be used on the X-API-Target-Name request header when performing and operation on this object. **Note:** This property is only returned when the BCPii interface was used for the request.

Description

This operation lists the capacity record for a given CPC.

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1379.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.

HTTP error status code	Reason code	Description
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Get Capacity Record Properties

The `Get Capacity Record Properties` operation retrieves the properties of a single Capacity Record designated by *{capacity-record-id}* from the CPC object designated by *{cpc-id}*. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/capacity-records/{capacity-record-id}
```

URI variables

Variable	Description
{cpc-id}	Object ID of the target CPC object.
{capacity-record-id}	Capacity Record identifier, returned by a previous List Capacity Records operation

Query parameters:

Name	Type	Rqd/Opt	Description
`properties`	List of String Enum	Optional	Filter string to limit returned properties to those that are identified here. This is a list of comma-separated strings where each string is a property name defined in the object's data model.

Response body contents

On successful completion, HTTP status code 200 (OK) is returned and the response body provides the current values of the properties for the Capacity Record as defined in "Data model" on page 1375.

Description

The URI path must designate an existing Capacity Record and the API user must have access permission to the associated CPC object. If either of these conditions is not met, status code 404 (Not Found) is returned.

On successful execution, HTTP status code 200 (OK) is returned and the response body contains all of the current properties as defined by the data model for the Capacity Record object.

Authorization requirements

This operation has the following authorization requirement:

- For the web services interface, object-access permission to the CPC object designated by *{cpc-id}*
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object designated by *{cpc-id}*.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in the response body contents section.

The following HTTP status codes are returned for the indicated errors, and the response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	276	The capacity record has expired, it can be deleted from the SE.
	302	The capacity record identifier in the URI (*{capacity-record-id}*) must be 1 to 8 characters.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
	274	The capacity record identifier in the URI (*{capacity-record-id}*) does not designate an existing capacity record.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
500 (Server Error)	275	An unexpected error occurred during the operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not communicating with the SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Inventory service data

Information about capacity records can be optionally included in the inventory data provided by the Inventory Service.

Inventory entries for the Capacity Record objects are included in the response to the Inventory Service's Get Inventory operation when the request specifies (explicitly by class, implicitly through a containing category, or by default) that objects of class **"cpc"** are to be included. An entry for a particular capacity record is included only if the API user has access permission to that object as described in the `Get Capacity Record Properties` operation.

For each Capacity Record object to be included, the inventory response array includes an entry that is a JSON object with the same contents as is specified in the response body contents section for "Get Capacity Record Properties" on page 1380. That is, the data provided is the same as would be provided if a `Get Capacity Record Properties` operation were requested targeting this object.

Chapter 12. Energy management

Energy Management is a management task that is pervasive and spread across several components in the systems management stack. Each layer in the stack needs to implement two key functions:

- A set of management functions appropriate for this level at the stack. Energy management functions provided by lower layers can be used to implement these functions.
- Management interfaces are provided that allows management layers above to configure and control the energy management functions.

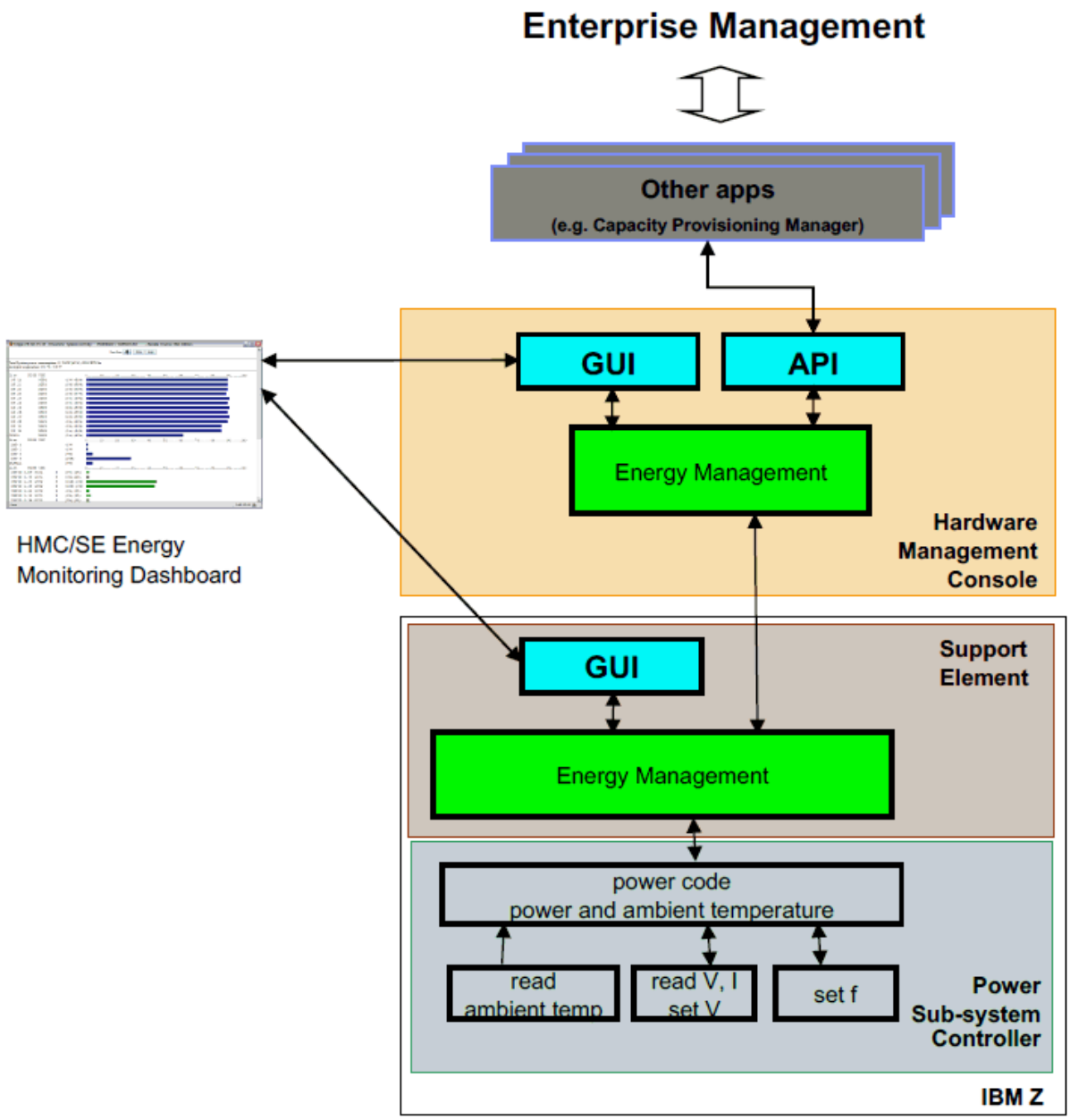

Figure 706. Energy management as applied throughout layers of enterprise management

To achieve this several pieces are needed:

© Copyright IBM Corp. 2022, 2024

Power and thermal monitoring

"You can't improve what you don't measure" is a trivial engineering paradigm. Measuring energy consumption and the thermal environment is key for management. Energy Monitoring for IBM zSystems was initially introduced with z9®.

Energy control

Based on the measurement data - either for an individual system or aggregated for a group of servers or even a complete data center - analytics can be implemented. These can keep a watch on given limits or can identify optimizing potentials. At a system level energy control mechanisms will be provided to allow for changing energy consumption of a system. These energy controls can be categorized into two groups:

- **Power saving** - Power saving mechanisms are used to reduce the average energy consumption of a system. Through powering off components or reducing performance the power saving is typically achieved. For older servers, such as the zEnterprise 196 the Static Power Savings Mode is implemented that reduces processor frequency and voltage for power saving purposes. Power saving capabilities are not supported on the IBM z15.
- **Power capping** - Power capping is a means to limit peak power consumption of a system. This is especially important in constrained data center environments. Today power and cooling allocation in data centers is usually done through the label power. This typically leads to a significant over-provisioning. Through power capping the power allocation for a system can be adjusted better to the real power consumption of a system and therefore more servers can be deployed within the same physical limits of their data center. Power capping is not supported on the IBM z14®, or the IBM z15.

Groups

A group is composed of an object that contains groups or another object and the object or objects it contains. For example, Figure 707 on page 1385 represents a CPC that contains a zCPC.

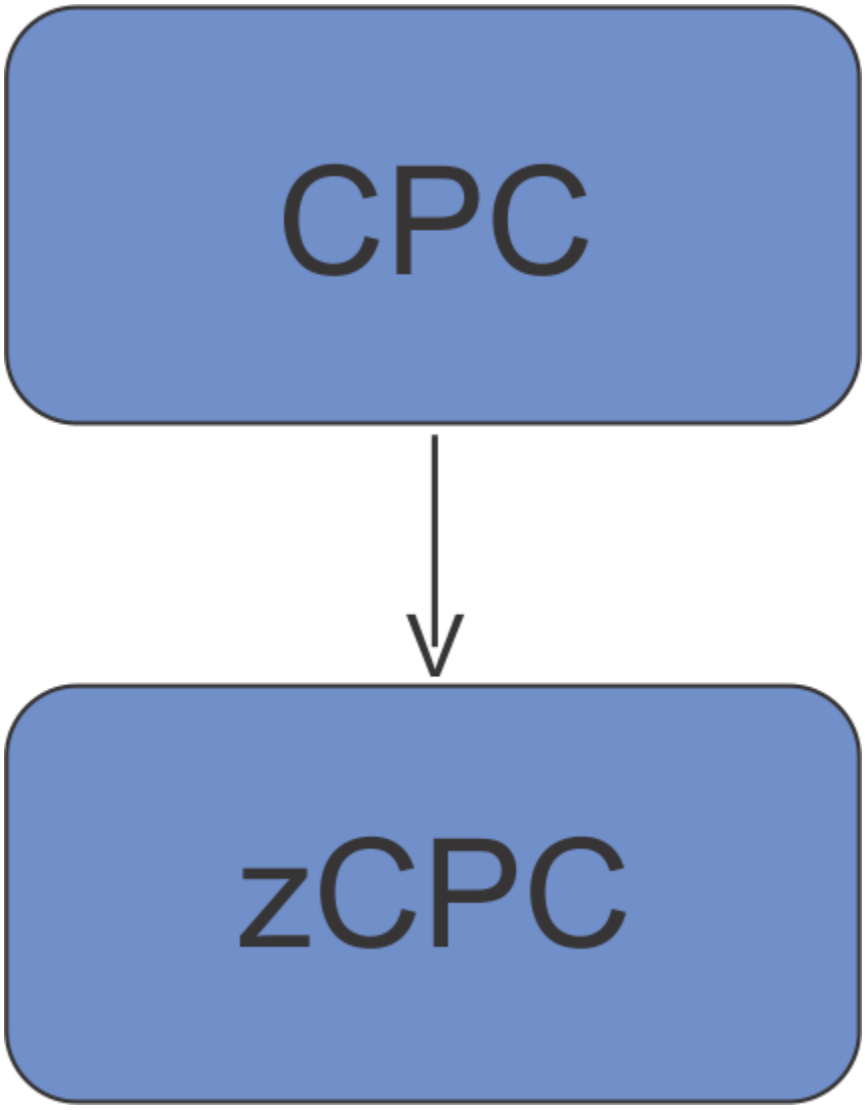

Figure 707. Example of a CPC group that contains a zCPC

Special states

In this chapter, the following states are used but the reasoning behind the states isn't always clear. So they are explained here in more detail:

"custom"
Occurs only on groups and indicates that the group does not control the children. Clients are able to alter the children of a group individually.

"under-group-control"
Occurs only on children and indicates that a group controls the state. When clients want to alter the state, the group must be set to **"custom"** first.

"not-supported"
Indicates that the feature (either power saving or power capping) is currently not supported, possible reasons can be:

- Hardware does not support it → permanent
- Firmware level does not support it → can change after a firmware update
- The hardware is not powered on → can change after the device is powered on.

"not-available"
Couldn't read the state of the underlying hardware.

"not-entitled"
Indicates that the automate feature is not installed and so power saving and power capping is not allowed.

Power saving

Power saving is a function that reduces the energy consumption of a system. Please note that power saving is only available if the Automate management enablement feature is installed. Power saving capabilities are not supported on the IBM z15. The possible settings include:

High performance
The power consumption and performance of the object are not reduced. This is the default setting.

Low power
The performance of the object is reduced to allow for low power consumption. When this setting is selected for CPC objects, all components of the object enabled for power saving have reduced performance to allow for low power consumption. Use this setting to enable group power saving.

Note: You can only set the power saving setting of the zCPC to Low power one time per calendar day in an air cooled system. This power save property is set to Not Supported if the current zCPC power saving setting is High performance but the zCPC has already entered Low power once within the calendar day.

Custom
Use Custom to disable group power saving and individually configure the components of the object for power saving.

Note: This setting is available only for CPC objects.

Group power saving

The following are important concepts regarding group power saving:

- Group power saving settings replace individual object settings--that is, the Power Saving setting of a CPC supersede the Power Saving setting of any object contained within the CPC.
- You can enable group power saving by setting the Power Saving setting of the CPC to Low power or High performance.
- You can change individual Power Saving settings only if the object is not under group power saving control.
- To disable group power saving without changing the individual Power Saving settings of the group members, change the Power Saving setting of the CPC to Custom.

Power capping

Please note that power capping is only available if the Automate management enablement feature is installed. Power capping is not supported on the IBM z14, or the IBM z15.

Group capping

The following are important concepts regarding group power capping:

- Group caps replace individual object caps—that is, the Cap Value of a CPC supersede the power cap of any object contained within the CPC.
- You can enable group capping by setting the Power Capping setting of the CPC to Enabled.
- You can change individual Cap Values if the object is not under group capping control.

- If a CPC contains an object that does not support power capping, the Power Rating is used in calculating the minimum power cap value for the group. The Power Rating can be found on the details window for an object.
- The maximum Cap Value for a group is the sum of the Power Rating of all Group objects.
- When a group component is powered off or removed, the group cap is redistributed to the remaining group components.
- To disable group capping without changing the individual power caps of the group members, change the Power Capping setting of the CPC to Custom.

Energy management operations summary

The following tables provide an overview of the operations provided. All POST operation are executed asynchronously and provide a job URI which can be used to obtain the current status of the asynchronous portion of the operation.

Note: The zCPC is not modeled as a full entity like a CPC, because energy management needs the zCPC to represent only IBM zSystems hardware without hardware extensions. That is the reason why all zCPC related operations are tied to the CPC.

Table 544. Energy management: operations summary

Operation name	HTTP method and URI path
"Set CPC Power Save" on page 1388	`POST /api/cpcs/{cpc-id}/operations/set-cpc-power-save`
"Set CPC Power Capping" on page 1390	`POST /api/cpcs/{cpc-id}/operations/set-cpc-power-capping`
"Get CPC Energy Management Data" on page 1397	`GET /api/cpcs/{cpc-id}/energy-management-data`
"Set zCPC Power Save" on page 1393	`POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-save`
"Set zCPC Power Capping" on page 1395	`POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-capping`
"Get Energy Optimization Advice Summary" on page 1399	`GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-summary`
"Get Energy Optimization Advice Details" on page 1402	`POST /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-details`

Table 545. Energy management: URI variables

Variable	Description
{cpc-id}	Object ID of a CPC

Energy Management for CPC object

The energy management for the CPC object represents all energy management for the CPC.

Data model

The data model for a CPC object includes some properties related to energy management. These properties are described in "Energy management related additional properties" on page 1026.

Operations

Set CPC Power Save

Use the `Set CPC Power Save` operation to set the power save setting of a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-cpc-power-save
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Name	Type	Rqd/Opt	Description
`power-saving`	String Enum	Required	The possible settings are: • **"high-performance"** - The power consumption and performance of the CPC are not reduced. This is the default setting. • **"low-power"** - Low power consumption for all components of the CPC enabled for power saving. • **"custom"** - Components may have their own settings changed individually. No component settings are actually changed when this mode is entered.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to control the average energy consumption of a CPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated CPC is under group control (see "Group capping" on page 1386) or the **cpc-power-saving** property of the CPC is set to **"not-supported"** or **"not-entitled"**. (See "Energy management related additional properties" on page 1026 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-saving settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of

the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described in "HTTP status and reason codes" on page 1389. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to all CPC and zCPC objects
  - Action/task permission to the **Power Save** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1388.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully

Job status codes	Job reason code	Description
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the IBM zSystems hardware
	162	Communication error occurred while trying to access the IBM zSystems hardware
	163	An error occurred at one or more children

If the job reason code is 163, the **job-results** field provided by the `Query Job Status` operation will contain an object with the following fields:

Field name	Type	Description
`errors`	Object array	A list of error objects, containing detailed error information about errors occurred on children
`at-least-one-operation-succeed`	Boolean	**True** indicates that the operation was successful for at least one child.

Each error object has this structure:

Job status codes	Job reason code	Description
`object-uri`	String URI	The canonical URI path for a specific object where the error occurred
`reason-code`	Integer	Specify the specific error type, possible values are: • 160 - A firmware error occurred while executing the operation • 161 - A hardware error occurred while performing the energy management operation • 162 - Communication error occurred while trying to access the hardware
`message`	String	A non-localized message provided for development purposes only. Client applications should not display this message directly to the user.

Set CPC Power Capping

Use the `Set CPC Power Capping` operation to set the power capping settings of a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-cpc-power-capping
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`power-capping-state`	String Enum	Required	The possible settings are: • **"disabled"** - The power cap of the CPC is not set and the peak power consumption is not limited. This is the default setting. • **"enabled"** - The peak power consumption of the CPC is limited to the current cap value. • **"custom"** - Individually configure the components of the IBM zSystems hardware for power capping. No component settings are actually changed when this mode is entered.
`power-cap-current`	Integer	Optional	Specifies the current cap value for the CPC in watts (W). The current cap value indicates the power budget for the CPC. This field is only required if the **power-capping-state** field is set to **"enabled"**. The **power-cap-current** must be between **cpc-power-cap-minimum** and **cpc-power-cap-maximum**: **cpc-power-cap-minimum** <= value <= **cpc-power-cap-maximum**

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to limit the peak power consumption of a CPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated CPC is under group control (see "Group capping" on page 1386) or the **cpc-power-capping-state** property of the CPC is set to **"not-supported"** or **"not-entitled"**. (See "Energy management related additional properties" on page 1026 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-capping settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described in "HTTP status and reason codes" on page 1392. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to all CPC and zCPC objects
  - Action/task permission to the **Power Capping** task.

- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in "Response body contents" on page 1391.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The **power-cap-current** field contains a value that is not in the range **cpc-power-cap-minimum** ... **cpc-power-cap-maximum**
	5	The **power-cap-current** field is not set, but **power-capping-state** field is set to **"enabled"**.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the IBM zSystems hardware
	162	Communication error occurred while trying to access the IBM zSystems hardware
	163	An error occurred at one or more children

If the job reason code is 163, the **job-results** field provided by the `Query Job Status` operation will contain an object with the following fields:

Field name	Type	Description
`errors`	Object array	A list of error objects, containing detailed error information about errors occurred on children
`at-least-one-operation-succeed`	Boolean	**True** indicates that the operation was successful for at least one child.

Each error object has this structure:

Job status codes	Job reason code	Description
`object-uri`	String URI	The canonical URI path for a specific object where the error occurred
`reason-code`	Integer	Specify the specific error type, possible values are: • 160 - A firmware error occurred while executing the operation • 161 - A hardware error occurred while performing the energy management operation • 162 - Communication error occurred while trying to access the hardware
`message`	String	A non localized message provided for development purposes only. Client applications should not display this message directly to the user.

Set zCPC Power Save

Use the `Set zCPC Power Save` operation to set the power save settings of the zCPC portion of a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-save
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`power-saving`	String Enum	Required	The possible settings are: • **"high-performance"** - The power consumption and performance of the zCPC are not reduced. This is the default setting. • **"low-power"** - Low power consumption for all components of the zCPC enabled for power saving.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to control the average energy consumption of a zCPC portion of the CPC *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated zCPC is under group control (see "Group capping" on page 1386) or the **zcpc-power-saving** property of the zCPC is set to **"not-supported"** or **"not-entitled"**. (See "Energy management related additional properties" on page 1026 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-saving settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described in "HTTP status and reason codes" on page 1394. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to all CPC and zCPC objects
  - Action/task permission to the **Power Save** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in"Response body contents" on page 1394.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.

HTTP error status code	Reason code	Description
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the IBM zSystems hardware
	162	Communication error occurred while trying to access the IBM zSystems hardware

Set zCPC Power Capping

Use the `Set zCPC Power Capping` operation to set the power capping settings of the zCPC portion of a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/set-zcpc-power-capping
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Request body contents

The request body is a JSON object with the following fields:

Field name	Type	Rqd/Opt	Description
`power-capping-state`	String Enum	Required	The possible settings are: • **"disabled"** - The power cap of the zCPC is not set and the peak power consumption is not limited. This is the default setting. • **"enabled"** - The peak power consumption of the zCPC is limited to the current cap value.

Field name	Type	Rqd/Opt	Description
`power-cap-current`	Integer	Optional	Specifies the current cap value for the zCPC in watts (W). The current cap value indicates the power budget for the zCPC. This field is only required if the **power-capping-state** field is set to **"enabled"**. The **power-cap-current** must be between **zcpc-power-cap-minimum** and **zcpc-power-cap-maximum**: **zcpc-power-cap-minimum** <= value <= **zcpc-power-cap-maximum**

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`job-uri`	String	URI of the asynchronous job that may be queried to retrieve status updates for action initiated by this operation.

Description

Use this operation to limit the peak power consumption of a zCPC object designated by *{cpc-id}*, or to remove a power consumption limit for this object. You can closely manage power allocations within the physical limits of your data center.

This operation will always fail if the designated zCPC is under group control (see "Group capping" on page 1386) or the **zcpc-power-capping-state** property of the zCPC is set to **"not-supported"** or **"not-entitled"**. (See "Energy management related additional properties" on page 1026 for details on this property.) In addition, this operation is only available if feature code 0020 is installed on the system.

The action to change the power-capping settings occurs asynchronously. If the request is accepted, an asynchronous job is initiated and an HTTP Status code of 202 (Accepted) is returned. The response body includes a URI that may be queried to retrieve the status of the asynchronous job. See the description of the `Query Job Status` operation for information on how to query job status. When the asynchronous job has completed, an asynchronous result message is sent, with Job status and reason codes described below. After completion, the `Query Job Status` operation may be used to retrieve the completion results.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to all CPC and zCPC objects
  - Action/task permission to the **Power Capping** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 202 (Accepted) is returned and the response body is provided as described in"Response body contents" on page 1396.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
	7	The **power-cap-current** field contains a value that is not in the range **zcpc-power-cap-minimum** ... **zcpc-power-cap-maximum**
	5	The **power-cap-current** field is not set, but **power-capping-state** field is set to **"enabled"**.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user is not authorized to access the object or perform this task.
	3	The server is not entitled to perform energy management.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	1	The operation cannot be performed because the object designated by the request URI is not in the correct state.
	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Job status and reason codes

Job status codes	Job reason code	Description
200 (OK)	N/A	Operation executed successfully
500 (Server Error)	160	A firmware error occurred while executing the operation
	161	A hardware error occurred while performing the operation on the IBM zSystems hardware
	162	Communication error occurred while trying to access the IBM zSystems hardware

Get CPC Energy Management Data

Use the `Get CPC Energy Management Data` operation to retrieve all energy management related data in one single call. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/energy-management-data
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`objects`	Array of objects	An array of nested em-data objects containing the energy management data. The format of each nested object is given in the next table.

Each nested em-data object contains the following fields:

Field name	Type	Description
`object-uri`	String/ URI	The canonical URI path of the specific object to which this em-data object pertains.
`object-id`	String	Object-id property of the specific object to which this em-data object pertains.
`class`	String	The type of the specific object to which this em-data object pertains.
`properties`	Object	Nested object containing the energy management properties for the object identified by the object-uri field, as described in the data model section for objects of the type indicated by the class field.
`error-occurred`	Boolean	If true, indicates that an error occurred while querying the data for the object specified by the object-uri. As a consequence the property could be null or incomplete.

Description

The `Get CPC Energy Management Data` is a convenience operation to allow a client to retrieve all energy management related data for a CPC in a single request rather than invoking several requests to retrieve this data.

Note that this operation returns data for a child object of the designated CPC only if the API user has object-access permission to that object. Children objects for which the API user does not have access are omitted from the response and no error is indicated.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface, object-access permission to the CPC object designated by the request, and for any children objects for which data is to be returned.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1398.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See"Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
404 (Not Found)	1	The object ID in the URI (*{cpc-id}*) does not designate an existing CPC object, or the API user does not have object-access permission to the object.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Get Energy Optimization Advice Summary

The `Get Energy Optimization Advice Summary` operation provides a summary of all currently available energy optimization advice for a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-summary
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC whose energy optimization advice summary is to be returned.

Response body contents

On successful completion, the response body is a JSON object with the following fields:

Field name	Type	Description
`advice`	Array of energy-optimization-advice-summary objects	A summary of all energy optimization advice available for the CPC. It is an array of nested energy-optimization-advice-summary objects as described in the next table.

Each nested energy-optimization-advice-summary object contains the following fields:

Field name	Type	Description
`type`	String Enum	Identifies the advice type. Valid values are: • **"environmental"** - Advice about environmental aspects of energy management such as air temperature. • **"processor-utilization"** - Advice related to the amount of processor utilization in the CPC.
`timestamp`	Timestamp	The time when this advice was created or updated. This field and the **last-energy-advice-time** property of the CPC are updated each time the **status** field is updated.
`summary`	String (0-153)	Human readable description of the **status** field.

Field name	Type	Description
`status`	String Enum	The current status of the parameters factored into this advice. The valid values for this field depend on the value of the **type** field. When **type** is **"environmental"**: • **"optimal"** - Parameters related to this advice are currently at optimal values; no changes are recommended at this time. • **"above-threshold"** - One or more of the parameters is above its threshold; changes are recommended. When **type** is **"processor-utilization"**: • **"low-utilization"** - System utilization is low. If static power save is enabled, power consumption can be reduced. • **"low-utilization-power-save"** - System utilization is low, but static power save is enabled. • **"high-utilization"** - System utilization is high and the system is configured optimally. • **"high-utilization-power-save"** - System utilization is high, but static power save is enabled. Performance can be increased by disabling static power save on the system.

Description

This operation returns summary information about the current energy optimization advice for the CPC object specified by *{cpc-id}*.

On successful execution, the energy optimization advice summary for each type of advice for which there is available data for the CPC is provided in the response body, and HTTP status code 200 (OK) is returned.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:
  - Object-access permission to the CPC object identified in the request URI.
  - Action/task permission to the **Energy Optimization Advisor** task.
- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1399.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 546. Get Energy Optimization Advice Summary: HTTP status and reason codes		
HTTP error status code	**Reason code**	**Description**
400 (Bad Request)	Various	Errors were detected during common request validation. See"Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing CPC, or it designates a CPC for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
GET /api/cpcs/a44d8ab8-e68e-3a07-9bff-e484a62ca00d/operations/get-energy-optimization-
   advice-summary HTTP/1.1
x-api-session: 2z7xak1rm55fqu2o48h5btjsu5rwqtexyuxrmvcn51rzbsxi6w
```

Figure 708. Get Energy Optimization Advice Summary: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 19:30:21 GMT
content-type: application/json;charset=UTF-8
content-length: 511
{
   "advice":[
      {
         "status":"above-threshold",
         "summary":"The ambient temperature is above the current threshold of 23.0\u00b0C
            (73.4\u00b0F). If the ambient temperature is lowered below the identified
            threshold, the system power consumption will be reduced by approximately 126 W.",
         "timestamp":1457031003347,
         "type":"environmental"
      },
      {
         "status":"high-utilization",
         "summary":"Based on processor utilization there are no system power consumption
            recommendations.",
         "timestamp":1448991133600,
         "type":"processor-utilization"
      }
   ]
}
```

Figure 709. Get Energy Optimization Advice Summary: Response

Get Energy Optimization Advice Details

The `Get Energy Optimization Advice Details` operation returns detailed information about currently available energy optimization advice for a CPC. This operation is supported using the BCPii interface.

HTTP method and URI

```
POST /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-details
```

In this request, the URI variable *{cpc-id}* is the object ID of the CPC whose energy optimization advice details are to be returned.

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
`timescale`	String Enum	Optional	Timescale for the advice graph data points. This is the amount of time to be covered by the data points in the graph. Valid values are: • **"four-hours"** • **"one-day"** • **"three-days"** • **"one-week"** If not specified, the default is **"one-day"**.

Field name	Type	Rqd/ Opt	Description
`type`	String Enum	Required	The type of advice details to return. Valid values are: • **"environmental"** - Advice about environmental aspects of energy management such as air temperature. • **"processor-utilization"** - Advice related to the processor utilization in the CPC.

Response body contents

On successful completion, the response body contains a JSON object that provides the values of the specific energy optimization advice object requested from the CPC. The advice object contains a set of fields common to all types of advice, and a set of additional fields that differ based on the advice type.

The common fields provided for all types of advice are defined in the following table:

Field name	Type	Description
`type`	String Enum	Identifies the advice type. Valid values are: • **"environmental"** - Advice about environmental aspects of energy management such as air temperature. • **"processor-utilization"** - Advice related to the processor utilization in the CPC.
`timestamp`	Timestamp	The time when this advice was created or updated. This field and the **last-energy-advice-time** property of the CPC are updated each time the **status** field is updated.
`summary`	String (0-153)	Human readable description of the **status** field.
`status`	String Enum	The current status of the parameters factored into this advice. The valid values for this field depend on the value of the **type** field. When **type** is **"environmental"**: • **"optimal"** - Parameters related to this advice are currently at optimal values; no changes are recommended at this time. • **"above-threshold"** - One or more of the parameters is above its threshold; changes are recommended. When **type** is **"processor-utilization"**: • **"low-utilization"** - System utilization is low. If static power save is enabled, power consumption can be reduced by the amount specified in the **power-saving** field. • **"low-utilization-power-save"** - System utilization is low, but static power save is enabled. • **"high-utilization"** - System utilization is high and the system is configured optimally. • **"high-utilization-power-save"** - System utilization is high, but static power save is enabled. Performance can be increased by disabling static power save on the system.

When the **type** field contains **"environmental"**, the response body also contains the following additional type-specific fields:

Field name	Type	Description
`ambient-temperature`	Float	The ambient temperature when this advice was generated, in degrees Celsius.
`wattage`	Integer	The electrical power usage when this advice was generated, in Watts.
`temperature-threshold`	Float	The temperature threshold that causes an increase of the system power consumption, in degrees Celsius.
`power-saving`	Integer	The amount of electrical power that can be saved when following this advice, in Watts.
`temperature-graph`	Array of float-data-point objects	An ordered collection of temperature data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is a float-data-point object representing the ambient environment temperature in degrees Celsius at a specific point in time. The interval is implicit given by the timestamps of the individual data points.
`wattage-graph`	Array of integer-data-point objects	An ordered collection of electrical power data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is an integer-data-point object representing the total system power, in Watts, at a specific point in time. The interval is implicit, given by the timestamps of the individual data points.

When the **type** field contains **"processor-utilization"**, the response body also contains the following additional type-specific fields:

Field name	Type	Description
`wattage`	Integer	The electrical power usage when this advice was generated, in Watts.
`power-saving`	Integer	The amount of electrical power that will be saved by following this advice, in Watts.
`utilization-threshold`	Integer (0-100)	The threshold in percentage. The threshold is used to calculate the current system status. If system utilization is below the threshold, enabling static power save should be considered.
`utilization`	Array of type-utilization objects	List of all processor types installed in the CPC and their percent utilization at the time this advice was created.
`static-power-save-recommended`	Boolean	Indicates whether enabling static power save is recommended. True if recommended; false otherwise.
`utilization-graphs`	Array of type-utilization -graph objects	A collection of processor utilization graphs. Each element of this array contains the utilization graph of a processor type available in the system.

Each nested float-data-point object contains the following fields:

Field name	Type	Description
`data`	Float	Floating point data value.
`timestamp`	Timestamp	The time when this data point was taken

Each nested integer-data-point object contains the following fields.

Field name	Type	Description
`data`	Integer	Integer data value.
`timestamp`	Timestamp	The time when this data point was taken.

Each type-utilization-graph object contains the following fields:

Field name	Type	Description
`type`	String Enum	The processor type. Valid values are: • **"cp"** - Central (general purpose) processor. • **"icf"** - Internal Coupling Facility processor. • **"ifl"** - Integrated Facility for Linux processor. • **"sap"** - Service assist processor. • **"sys"** - Total system utilization. (Used for threshold monitoring).
`utilization-graph`	Array of integer-data-point objects	An ordered collection of utilization data points, in descending time order, beginning with the most recent, covering as much of the requested time period as is available. Each element of this array is an integer-data-point object representing the utilization of the processor type in percentage at a specific point in time. The interval is implicit, given by the timestamps of the individual data points.

Each nested type-utilization object contains the following fields:

Field name	Type	Description
`type`	String Enum	The processor type. Valid values are: • **"cp"** - Central (general purpose) processor. • **"icf"** - Internal Coupling Facility processor. • **"ifl"** - Integrated Facility for Linux processor. • **"sap"** - Service assist processor. • **"sys"** - Total system utilization. (Used for threshold monitoring).
`utilization`	Integer (0-100)	The current utilization in percentage for the given processor type.

Description

This operation returns detailed information about the requested type of energy optimization advice for the CPC object specified by *{cpc-id}*.

On successful execution, the energy optimization advice details for the requested type of advice for the CPC are provided in the response body, and HTTP status code 200 (OK) is returned. The details cover as much of the requested time period as is available.

If the request URI does not identify a CPC object to which the API user has object-access permission, HTTP status code 404 (Not Found) is returned.

Authorization requirements

This operation has the following authorization requirements:

- For the web services interface:

- Object-access permission to the CPC object identified in the request URI.
- Action/task permission to the **Energy Optimization Advisor** task.

- For the BCPii interface, the source partition must have receive BCPii security controls permissions for the CPC object.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1403.

Otherwise, the following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and associated error message.

Table 547. Get Energy Optimization Advice Details: HTTP status and reason codes

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See"Common request validation reason codes" on page 66 for a list of the possible reason codes.
403 (Forbidden)	0	The request used the BCPii interface and the source partition does not have receive BCPii security controls permission for the CPC object.
	1	The user under which the API request was authenticated does not have the required authority to perform the requested action.
404 (Not Found)	1	The request URI does not designate an existing CPC, or it designates a CPC for which the API user does not have object-access permission.
	4	The object designated by the request URI does not support the requested operation.
409 (Conflict)	329	The operation cannot be performed because the CPC designated by the request URI is an unmanaged CPC, which is not supported by this operation.
503 (Service Unavailable)	1	The request could not be processed because the HMC is not currently communicating with an SE needed to perform the requested operation.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Example HTTP interaction

```
POST /api/cpcs/a44d8ab8-e68e-3a07-9bff-e484a62ca00d/operations/get-energy-optimization-
   advice-details HTTP/1.1
x-api-session: 2szwxczuje9cttj01145ozlab65x9osmpvvdibob5w0nvgwm80
content-type: application/json
content-length: 52
{
   "timescale":"four-hours",
   "type":"environmental"
}
```

Figure 710. Get Energy Optimization Advice Details: Request

```
200 OK
server: zSeries management console API web server / 2.0
cache-control: no-cache
date: Tue, 08 Mar 2016 19:32:27 GMT
content-type: application/json;charset=UTF-8
content-length: 2683
{
   "ambient-temperature":24.200000762939453,
   "power-saving":126,
   "status":"above-threshold",
   "summary":"The ambient temperature is above the current threshold of 23.0\u00b0C
      (73.4\u00b0F). If the ambient temperature is lowered below the identified threshold,
      the system power consumption will be reduced by approximately 126 W.",
   "temperature-graph":[
      {
         "data":23.80500030517578,
         "timestamp":1457451614605
      },
      {
         "data":23.577499389648438,
         "timestamp":1457452214606
      },
      {
         "data":23.645000457763672,
         "timestamp":1457452814605
      },
      {
         "data":24.00749969482422,
         "timestamp":1457453414605
      },
      {
         "data":23.850000381469727,
         "timestamp":1457454014605
      },
      {
         "data":23.697500228881836,
         "timestamp":1457454614605
      },
      {
         "data":23.704999923706055,
         "timestamp":1457455214605
      },
      {
         "data":23.825000762939453,
         "timestamp":1457455814605
      },
      {
         "data":23.704999923706055,
         "timestamp":1457456414605
      },
      {
         "data":23.75,
         "timestamp":1457457014605
      },
```

Figure 711. Get Energy Optimization Advice Details: Response (Part 1)

```
        {
           "data":23.704999923706055,
           "timestamp":1457457614605
        },
        {
           "data":23.7450008392334,
           "timestamp":1457458214605
        },
        {
           "data":23.78499984741211,
           "timestamp":1457458814606
        },
        {
           "data":23.834999084472656,
           "timestamp":1457459414607
        },
        {
           "data":23.795000076293945,
           "timestamp":1457460014606
        },
        {
           "data":23.912500381469727,
           "timestamp":1457460614604
        },
        {
           "data":23.80500030517578,
           "timestamp":1457461214606
        },
        {
           "data":23.6875,
           "timestamp":1457461814606
        },
        {
           "data":23.799999237060547,
           "timestamp":1457462414604
        },
        {
           "data":23.889999389648438,
           "timestamp":1457463014605
        },
        {
           "data":23.907499313354492,
           "timestamp":1457463614604
        },
        {
           "data":23.877500534057617,
           "timestamp":1457464214614
        },
        {
           "data":24.012500762939453,
           "timestamp":1457464814605
        },
```


Figure 712. Get Energy Optimization Advice Details: Response (Part 2)

```
        {
            "data":24.200000762939453,
            "timestamp":1457465414604
        }
    ],
    "temperature-threshold":23.0,
    "timestamp":1457031003347,
    "type":"environmental",
    "wattage":6952,
    "wattage-graph":[
        {
            "data":6957,
            "timestamp":1457451614605
        },
        {
            "data":6970,
            "timestamp":1457452214606
        },
        {
            "data":6968,
            "timestamp":1457452814605
        },
        {
            "data":6970,
            "timestamp":1457453414605
        },
        {
            "data":6965,
            "timestamp":1457454014605
        },
        {
            "data":6970,
            "timestamp":1457454614605
        },
        {
            "data":6965,
            "timestamp":1457455214605
        },
        {
            "data":6963,
            "timestamp":1457455814605
        },
        {
            "data":6958,
            "timestamp":1457456414605
        },
        {
            "data":6969,
            "timestamp":1457457014605
        },
        {
            "data":6963,
            "timestamp":1457457614605
        },
```


Figure 713. Get Energy Optimization Advice Details: Response (Part 3)

```
        {
           "data":6963,
           "timestamp":1457458214605
        },
        {
           "data":6967,
           "timestamp":1457458814606
        },
        {
           "data":6961,
           "timestamp":1457459414607
        },
        {
           "data":6967,
           "timestamp":1457460014606
        },
        {
           "data":6963,
           "timestamp":1457460614604
        },
        {
           "data":6968,
           "timestamp":1457461214606
        },
        {
           "data":6969,
           "timestamp":1457461814606
        },
        {
           "data":6964,
           "timestamp":1457462414604
        },
        {
           "data":6963,
           "timestamp":1457463014605
        },
        {
           "data":6951,
           "timestamp":1457463614604
        },
        {
           "data":6941,
           "timestamp":1457464214614
        },
        {
           "data":6947,
           "timestamp":1457464814605
        },
        {
           "data":6955,
           "timestamp":1457465414604
        }
    ]
}
```


Figure 714. Get Energy Optimization Advice Details: Response (Part 4)

Appendix A. Base Control Program internal interface (BCPii)

In addition to the HTTP based web services interface, starting with version 2.15.0 of the Hardware Management Console and Support Element a new BCPii interface is available to the APIs described in this document. The client side of the BCPii interface is the HWIREST interface exposed by z/OS. For more information, see *z/OS MVS Callable Services for HLL*.

Similar to the web services interface, the BCPii interface is also a request-and-response oriented programming interface by which z/OS client applications obtain information about the system resources managed by zManager, and by which those applications can perform provisioning, configuration or control actions on those resources. This interface has also been influenced by the Representational State Transfer (REST) style of interface design, which uses the same URIs and methods as the web services interface.

Unlike the web services interface, the BCPii interface does not flow standard HTTP protocol over TCP/IP connection, instead it uses internal communications paths between the z/OS operating system and the hosting hardware. This means that the entry point for the BCPii interface is a specific hardware system, whereas the entry point for the web services interface is a Hardware Management Console. The scope of management for the web services interface is the target Hardware Management Console and the systems and resources being managed by it, while the scope for the BCPii interface is the hosting system and its resources and any other system and associated resources being managed by the Hardware Management Consoles that are managing the hosting system.

Supported objects and operations

The BCPii interface is focused on the data and operations for the Core IBM zSystems resources. Currently, only the BCPii application that issued an asynchronous request will have the information to POLL for its result, so BCPii applications should continue to use the existing HWIEVENT service and ENF exit to learn of other types of CPC and LPAR events. Following is the complete set of objects and operations that are available using the BCPii interface and their corresponding authority requirements.

In addition to any authorization details mentioned by a specific REST API, an application taking advantage of the new BCPii interface needs to have the appropriate authority to the particular resource that it is trying to access. This authority is granted by defining the corresponding profile in the FACILITY resource class and permitting the user ID, under which BCPii application is executed, the appropriate access level to the profile. In addition to the FACILITY Class Profile, each HWIREST request, unless noted otherwise, requires an X-API-Target-Name header which *resembles* the FACILITY Class Profile. The content of the X-API-Target-Name is the value of the **target-name** property returned by the Support Element for a specific object and is used on all corresponding interactions with that object.

“Session management services” on page 112:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Query API Version`
 	
`GET /api/version`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

“Asynchronous job processing” on page 151:

© Copyright IBM Corp. 2022, 2024

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Query Job Status`
 	
`GET /api/jobs/{job-id}`
	netid.nau OR *netid.nau.imagename*	HWI.TARGET.*netid.nau* OR HWI.TARGET.*netid.nau.imagename*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
				
				The call should re-use the target name associated with the originating operation that resulted in this job URI, which may have been either against a CPC or an LPAR.

`Delete Completed Job Status`
 	
`DELETE /api/jobs/{job-id}`
	netid.nau OR *netid.nau.imagename*	HWI.TARGET.*netid.nau* OR HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
				
				The call should re-use the target name associated with the originating operation that resulted in this job URI, which may have been either against a CPC or an LPAR.

`Cancel Job`[1] `POST /api/jobs/{job-id}/operations/cancel`	*netid.nau* OR *netid.nau.imagename*	HWI.TARGET.*netid.nau* OR HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name. The call should re-use the target name associated with the originating operation that resulted in this job URI, which may have been either against a CPC or an LPAR.
[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

“Console object” on page 801:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Get Console Properties`
 	
`GET /api/console`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Restart Console`[1] `POST /api/console/operations/restart`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.
`Shutdown Console`[1] `POST /api/console/operations/shutdown`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Get Console Audit Log`
 	
`GET /api/console/operations/get-audit-log`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Get Console Security Log`
 	
`GET /api/console/operations/get-security-log`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Get Console Events Log`
 	
`GET /api/console/operations/get-events-log`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`List Hardware Messages`
 	
`GET /api/console/hardware-messages`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Get Console Hardware Message Properties`
 	
`GET /api/console/hardware-messages/{hardwaremessage-id}`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Delete Console Hardware Message`
 	
`DELETE /api/console/hardware-messages/{hardwaremessage-id}`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Request Console Service`[1] `POST /api/console/hardware-messages/{hardware-message-id}/operations/request-service`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Get Console Service Request Information`
 	
`GET /api/console/hardware-messages/{hardware-message-id}/operations/get-service-information`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Decline Console Service`[1] `POST /api/console/hardware-messages/{hardware-message-id}/operations/decline-service`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`List Permitted Adapters`
 	
`GET /api/console/operations/list-permitted-adapters`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.
				
				[Added by feature **adapter-network-information**]

[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

“Group Object” on page 996:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`List Custom Groups`
 	
`GET /api/groups`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Get Custom Group Properties`
 	
`GET /api/groups/{group-id}`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Create Custom Group`
 	
`POST /api/groups`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Delete Custom Group`
 	
`DELETE /api/groups/{group-id}`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`Add Member to Custom Group`[1] `POST /api/groups/{group-id}/operations/add-member`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.
`Remove Member from Custom Group`[1] `POST /api/groups/{group-id}/operations/remove-member`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

`List Custom Group Members`
 	
`GET /api/groups/{group-id}/members`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the console is associated with.

[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

“CPC object” on page 1010:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`List CPC Objects`
 	
`GET /api/cpcs`
	N/A	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
				
				For List of CPCs the X-API-Target-Name is not specified, however the response will only include CPCs for which the user is authorized. That is, the user must be permitted to the RACF profile that corresponds to that specific CPC in order for that CPC to be returned as part of the response.

`Get CPC Properties`
 	
`GET /api/cpcs/{cpc-id}`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

`Update CPC Properties`
 	
`POST /api/cpcs/{cpc-id}`
	netid.nau	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
`Activate CPC`[1] `POST /api/cpcs/{cpc-id}/operations/activate`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Deactivate CPC`[1] `POST /api/cpcs/{cpc-id}/operations/deactivate`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Import Profiles`[1] `POST /api/cpcs/{cpc-id}/operations/import-profiles`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Export Profiles`[1] `POST /api/cpcs/{cpc-id}/operations/export-profiles`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Add Temporary Capacity`[1] `POST /api/cpcs/{cpc-id}/operations/add-temp-capacity`	*netid.nau*	HWI.TARGET.*netid.nauu*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Remove Temporary Capacity`[1] `POST /api/cpcs/{cpc-id}/operations/remove-temp-capacity`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Swap Current Time Server`[1] `POST /api/cpcs/{cpc-id}/operations/swap-cts`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Set STP Configuration`[1] `POST /api/cpcs/{cpc-id}/operations/set-stp-config`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Change STP-only Coordinated Timing Network`[1] `POST /api/cpcs/{cpc-id}/operations/change-stponly-ctn`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Join STP-only Coordinated Timing Network`[1] `POST /api/cpcs/{cpc-id}/operations/join-stponly-ctn`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
`Leave STP-only Coordinated Timing Network`[1] `POST /api/cpcs/{cpc-id}/operations/leave-stponly-ctn`	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
List CPC Hardware Messages GET /api/cpcs/*{cpc-id}*/hardware-messages	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get CPC Hardware Message Properties GET /api/cpcs/*{cpc-id}*/hardware-messages/*{hardware-message-id}*	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Delete CPC Hardware Message DELETE /api/cpcs/*{cpc-id}*/hardware-messages/*{hardware-message-id}*	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Request CPC Service[1] POST /api/cpcs/*{cpc-id}*/hardware-messages/*{hardware-message-id}*/operations/request-service	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get CPC Service Request Information GET /api/cpcs/*{cpc-id}*/hardware-messages/*{hardware-message-id}*/operations/get-service-information	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Decline CPC Service[1] POST /api/cpcs/*{cpc-id}*/hardware-messages/*{hardware-message-id}*/operations/decline-service	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get CPC Audit Log GET /api/cpcs/*{cpc-id}*/operations/get-audit-log	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get CPC Security Log GET /api/cpcs/*{cpc-id}*/operations/get-security-log	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get CPC Events Log GET /api/cpcs/*{cpc-id}*/operations/get-event-log	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get LPAR Resource Assignments GET /api/cpcs/*{cpc-id}*/operations/get-lpar-resource-assignments	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Get LPAR Controls`
 	
`GET /api/cpcs/{cpc-id}/operations/get-lpar-controls`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

`Update LPAR Controls`
 	
`POST /api/cpcs/{cpc-id}/operations/update-lpar-controls`
	netid.nau	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

`Import CPC Certificate`
 	
`POST /api/cpcs/{cpc-id}/operations/import-certificate`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
				
				[Added by feature **secure-boot-with-certificates**]

`Report a CPC Problem`
 	
`POST /api/cpcs/{cpc-id}/operations/report-problem`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
				
				[Added by feature **report-a-problem**]

`Get CPC Historical Sustainability Data`
 	
`POST /api/cpcs/{cpc-id}/operations/get-historical-sustainability-data`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where netid.nau represents the 3– to 17–character SNA name of the particular CPC.
				
				[Added by feature **environmental-metrics**]

[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

“Logical Partition object” on page 1169:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`List Logical Partitions of a CPC`
 	
`GET /api/cpcs/{cpc-id}/logical-partitions`
	netid.nau	HWI.TARGET.*netid.nau* AND HWI.TARGET.*netid.nau.imagename*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
				
				For List of LPARs the X-API-Target-Name reflects the CPC the LPARs are associated with (*netid.nau*).
				
				The response itself is limited to LPARs that the user is authorized to. That is, the user must be permitted to the RACF profile that corresponds to that specific LPAR (HWI.TARGET.*netid.nau.imagename*) in order for that LPAR to appear in the List response in addition to the RACF profile (HWI.TARGET.*netid.nau*) that corresponds to the source CPC.

`List Permitted Logical Partitions`
 	
`GET /api/console/operations/list-permitted-logical-partitions`
	netid.nau	HWI.TARGET.*netid.nau.imagename*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
Get Logical Partition Properties GET /api/logical-partitions/*{logical-partition-id}*	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Update Logical Partition Properties POST /api/logical-partitions/*{logical-partition-id}*	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Activate Logical Partition[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/activate	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Deactivate Logical Partition[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/deactivate	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Reset Normal[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/reset-normal	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Reset Clear[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/reset-clear	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Load Logical Partition[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/load	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
PSW Restart[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/psw-restart	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
Start Logical Partition[1] POST /api/logical-partitions/*{logical-partition-id}*/operations/start	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
`Stop Logical Partition`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/stop`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
`Send OS Command`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/send-os-cmd`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.

`List OS Message of a Logical Partition`
 	
`GET /api/logical-partitions/`*`{logical-partition-id}`*`/operations/list-os-messages`
	netid.nau.imagename	HWI.TARGET.*netid.nau.imagename*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.

`Delete Logical Partition OS Message`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/delete-os-message`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
`SCSI Load`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/scsi-load`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
`SCSI Dump`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/scsi-dump`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
`NVMe Load`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/nvme-load`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.
`NVMe Dump`[1] `POST /api/logical-partitions/`*`{logical-partition-id}`*`/operations/nvme-dump`	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
Assign Certificate to Logical Partition POST /api/logical-partitions/*{logical-partition-id}*/operations/assign-certificate	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name. [Added by feature **secure-boot-with-certificates**]
Unassign Certificate from Logical Partition POST /api/logical-partitions/*{logical-partition-id}*/operations/unassign-certificate	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name. [Added by feature **secure-boot-with-certificates**]
Report a Logical Partition Problem POST /api/logical-partitions/*{logical-partition-id}*/operations/report-problem	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name. [Added by feature **report-a-problem**]
Get Logical Partition Historical Sustainability Data POST /api/logical-partitions/*{logical-partition-id}*/operations/get-historical-sustainability-data	*netid.nau.imagename*	HWI.TARGET.*netid.nau.imagename*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *imagename* represents the 1– to 8-character LPAR name. [Added by feature **environmental-metrics**]
[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

"Certificate object" on page 1266: [Added by feature **secure-boot-with-certificates**]

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
Delete Certificate DELETE /api/certificates/*{certificate-id}*	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get Certificate Properties GET /api/certificates/*{certificate-id}*	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
Get Encoded Certificate GET /api/certificates/*{certificate-id}*/operations/get-encoded	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.
List Certificates GET /api/certificates	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Update Certificate Properties`
 	
`POST /api/certificates/{certificate-id}`
	netid.nau	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC.

"Reset activation profile" on page 1278:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Create Reset Activation Profile`
 	
`POST /api/cpcs/{cpc-id}/reset-activation-profiles`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

`Delete Reset Activation Profile`
 	
`DELETE /api/cpcs/{cpc-id}/reset-activation-profiles/{reset-activation-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

"Image activation profile" on page 1291:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`List Image Activation Profiles`
 	
`GET /api/cpcs/{cpc-id}/image-activation-profiles`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the activation profiles are defined on.
				
				For List of image activation profiles the X-API-Target-Name reflects the CPC the image activation profiles are associated with (*netid.nau*).

`Get Image Activation Profile Properties`
 	
`GET /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.

`Update Image Activation Profile Properties`
 	
`POST /api/cpcs/{cpc-id}/image-activation-profiles/{image-activation-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
Assign Certificate to Image Activation Profile POST /api/cpcs/*{cpc-id}*/image-activation-profiles/*{image-activation-profile-name}*/operations/assign-certificate	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with. [Added by feature **secure-boot-with-certificates**]
Unassign Certificate from Image Activation Profile POST /api/cpcs/*{cpc-id}*/image-activation-profiles/*{image-activation-profile-name}*/operations/unassign-certificate	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with. [Added by feature **secure-boot-with-certificates**]
Create Image Activation Profile POST /api/cpcs/*{cpc-id}*/image-activation-profiles	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with. [Added by feature **create-delete-activation-profiles**]
Delete Image Activation Profile DELETE /api/cpcs/*{cpc-id}*/image-activation-profiles/*{image-activation-profile-name}*	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with. [Added by feature **create-delete-activation-profiles**]

“Load activation profile” on page 1340:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
List Load Activation Profiles GET /api/cpcs/*{cpc-id}*/load-activation-profiles	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the activation profiles are defined on. For List of load activation profiles the X-API-Target-Name reflects the CPC the load activation profiles are associated with (*netid.nau*).
Get Load Activation Profile Properties GET /api/cpcs/*{cpc-id}*/load-activation-profiles/*{load-activation-profile-name}*	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
Update Load Activation Profile Properties POST /api/cpcs/*{cpc-id}*/load-activation-profiles/*{load-activation-profile-name}*	*netid.nau*	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Create Load Activation Profile`
 	
`POST /api/cpcs/{cpc-id}/load-activation-profiles`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

`Delete Load Activation Profile`
 	
`DELETE /api/cpcs/{cpc-id}/load-activation-profiles/{load-activation-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

"Group profile" on page 1362:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`List Group Profiles`
 	
`GET /api/cpcs/{cpc-id}/group-profiles`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the group profiles are defined on.
				
				For List of group profiles the X-API-Target-Name reflects the CPC the group profiles are associated with (*netid.nau*).

`Get Group Profile Properties`
 	
`GET /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the group profile is defined.

`Update Group Profile Properties`
 	
`POST /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	UPDATE	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the group profile is defined.

`Create Group Profile`
 	
`POST /api/cpcs/{cpc-id}/group-profiles`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

`Delete Group Profile`
 	
`DELETE /api/cpcs/{cpc-id}/group-profiles/{group-profile-name}`
	netid.nau	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC the profile is associated with.
				
				[Added by feature **create-delete-activation-profiles**]

"Capacity records" on page 1375:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
List Capacity Records GET /api/cpcs/*{cpc-id}*/capacity-records	*netid.nau*	HWI.TARGET.*netid.nau* AND HWI.CAPREC.*netid.nau.caprec*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *caprec* represents an 8–character capacity record name. For List of capacity records the X-API-Target-Name reflects the CPC the capacity records are associated with (*netid.nau*). The response itself is limited to capacity records that the user is authorized to. That is, the user must be permitted to the RACF profile that corresponds to that specific capacity record (HWI.CAPREC.*netid.nau.caprec)* in order for that capacity record to appear in the List response in addition to the RACF profile (HWI.TARGET.*netid.nau*) that corresponds to the source CPC.
Get Capacity Record Properties GET /api/cpcs/*{cpc-id}*/capacity-records/*{capacity-record-id}*	netid.nau.caprec	HWI.CAPREC.*netid.nau.caprec*	READ	Where *netid.nau* represents the 3– to 17–character SNA name of the particular CPC and *caprec* represents an 8–character capacity record name.

"Energy Management for CPC object" on page 1387:

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description
Set CPC Power Save[1] POST /api/cpcs/*{cpc-id}*/operations/set-cpc-power-save	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.
Set CPC Power Capping[1] POST /api/cpcs/*{cpc-id}*/operations/set-cpc-power-capping	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.
Set zCPC Power Save[1] POST /api/cpcs/*{cpc-id}*/operations/set-zcpc-power-save	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.
Set zCPC Power Capping[1] POST /api/cpcs/*{cpc-id}*/operations/set-zcpc-powercapping	*netid.nau*	HWI.TARGET.*netid.nau*	CONTROL	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.
Get CPC Energy Management Data GET /api/cpcs/*{cpc-id}*/energy-management-data	*netid.nau*	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.

Action	Request X-API-Target-Name	FACILITY Class Profile	Minimum Access	Description

`Get Energy Optimization Advice Summary`
 	
`GET /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-summary`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.

`Get Energy Optimization Advice Details`
 	
`POST /api/cpcs/{cpc-id}/operations/get-energy-optimization-advice-details`
	netid.nau	HWI.TARGET.*netid.nau*	READ	Where *netid.nau* represents the 3- to 17-character SNA name of the particular CPC.

[1]This operation is not permitted for BCPii REXX execs running in the TSO/E environment or an ISV-provided REXX environment unless z/OS is version 2.4 or 2.5 with APAR OA61976 applied or a later z/OS version.				

Security controls

Security controls exist to configure additional permissions for API requests that use the BCPii interface. These security controls work in concert with the RACF profile checking performed by z/OS allow for controlling the objects and operations for these objects that are available to BCPii applications.

The following additional capabilities can be configured.

- Enable the logical partition to send commands
  - Allows for control over which logical partitions (i.e. operating systems) can send BCPii requests. The value of this setting can be changed using the **Change LPAR Security** task to affect the current value for an active logical partition or through the **Image Activation Profile** task to affect the value to be used the next time the logical partition is activated.
- Enable the partition to receive commands from other logical partitions
  - Allows for control over whether requests targeting a specific object should be received and handled. These controls are available for CPC and Logical Partition. The **System Details** task can be used to configure these controls for a CPC and the **Change LPAR Security** task can be used to configure these controls for an active Logical Partition. Additionally, the **Image Activation Profile** task allows for these controls to be configured to define the permissions that should be used when a Logical Partition is activated.

    In all cases, it is possible to disallow requests completely for a target object. Conversely, it is also possible to allow requests from any source to be allowed for a target object. It is also possible to only allow requests to from a specific set of sources.

The console help system provides additional information about the tasks used to configure these security controls.

Special considerations for BCPii

While the new BCPii interface is REST-like and supports the same methods and URIs as the web services interface, there are some special considerations to consider. Such as:

- request targeting
- character set conversion
- content type
- chunked encoding

- max content length [Added by feature **rc-409-15**]

Request targeting:

For the web services interface the target of a request is obvious; it is the HMC on the other end of the HTTP connection. For the BCPii interface it is not as obvious. The entry point into the HMC and SE network is the SE associated with the system hosting the source z/OS operating system. All BCPii requests are initially handled by this *local* SE. The *local* SE then establishes a secure HTTP connection with the target SE and the HTTP request/response flow over this connection, with the response data being sent back to the originating z/OS operating system.

The BCPii application needs to inform the *local* SE about where to route an incoming request. This is accomplished through the X-API-Target-Name header. This header is used by z/OS BCPii to perform RACF profile checks, but it also contains the information for the name of the target SE. The *local* SE uses part of the X-API-Target-Name header value to determine the name of the target SE and then using routing information shared by the HMC with the SE, a secure HTTP connection is used to forward the request to the correct SE for handling. This is even true for requests targeting the *local* SE by using a local HTTP connection.

The traffic that flows between SEs flows through an HMC. This means that two SEs need to be managed by at least one common HMC for traffic to be routed between them. These HMCs need to be configured as *Change Management* HMCs, since only *Change Management* HMCs can perform this routing function. For reliability it makes sense to have more than one HMC configured that can route between two SEs. It is also important to note that this routing is not at the network level so these HMCs are not acting as a general network router.

The BCPii application should never compose the targeting information itself but should reuse the target information that was returned to it when it retrieved information for a specific resource.

Each application is expected to begin by issuing the `List CPC Objects` operation (see "List CPC Objects" on page 1036) to obtain URI and targeting information for the CPCs it's going to interact with. The `List CPC Objects` operation does not require targeting information and is always directed to the *local* SE.

Character set conversion:

The z/OS BCPii programming interfaces provide the ability for the application to specify the character set being used for the data used for a given request. This means that the method, URI, headers, and any request body data will use the specified character set. It also means that the response headers and response body data will use this character set. This allows a z/OS application to use a more natural character EBCDIC character set, such as IBM-1047. The *local* SE will convert the request information to UTF-8 internally before that data is used for the HTTP request to the target SE and then convert any response headers and body back into the application specified character set before sending them back to the source z/OS operating system.

Content type and chunked encoding:

With very few exceptions the content type for request and response data is application/JSON for the operations described in this book. This is the case for all operations supported for the BCPii interface. With this in mind, the *Content-Type* header is not allowed to be specified for BCPii requests and is not returned for the corresponding responses. The BCPii application needs to provide any request body content as JSON using the character set specified on the request. Similarly, any response body data will be JSON using the character set specified on the request.

There are some operations that are documented to return the response body using *chunked* encoding. Any of these operations that are also supported for the BCPii interface, will **not** use *chunked* encoding for the response. Instead the *local* SE will deal with de-chunking the response body data so that the BCPii application does not need to do it.

Max content length:

The z/OS BCPii programming interface requires applications to specify the maximum allowable content length of the response body. This specification is beyond the scope of this document, this section will only

detail the implications for the bridge between a BCPii request/response and its HTTP counterpart. [Added by feature **rc-409-15**]

There is not a concept for specifying a maximum allowable content length on normal Web Services API HTTP requests. A BCPii request will be built into an HTTP request to be sent and processed by the target system without any consideration for the maximum content length. However, the content length of the HTTP response will be checked against the specified maximum content length when building the BCPii response to send back to the originating z/OS operating system. If the size of the response body for the Web Services API request exceeds the maximum allowable content length a conflict has occurred. It is important to note that this conflict occurs after the target system has processed the request, and as such the request may have been successful but the response data was unable to be returned to the client application.

When such a conflict occurs a 409 (Conflict) status with a response body (see "Common request validation reason codes" on page 66) containing reason code 15 will be returned. The **error-details** field of the error response body will contain the following object:

Field name	Type	Description
undeliverable-response	undeliverable-response-info object	An undeliverable-response-info object (described in the next table) containing information on the response that was unable to be delivered to the client.

The undeliverable-response-info object contains the following fields:

Field name	Type	Description
http-status	Integer	Specifies the HTTP status code.
content-length	Integer	Specifies the length of the response body.

Any error response body that exceeds the maximum allowable content length will be shortened to ensure it will fit. This holds true for the aforementioned HTTP status 409 with reason code 15 response as well as any other standard error response that was returned by the target system. If this occurs, the **bcpii-error**, **http-status**, **reason**, **message** and **error-details** fields are prioritized, and the response will always be well-formed JSON.

Configuring the Support Element for BCPii

The necessary Support Element configuration steps can be performed by using the **Customize API Settings** task.

To configure the Support Element:

1. Log on to the Console in *Access Administrator* mode.
2. Select **Tasks Index** in the left navigation pane.
3. Select the **Customize API Settings** task from the tasks list.
4. Select **Enable**.
5. Specify any **SNMP agent parameters** desired. **Note:** No special SNMP agent parameters are required for the API to work correctly.
6. Add one entry in the **Community Names** box by selecting **Add** push button to add a new community name or select the **Change** push button to change an existing community name.

   Name
   This field should be filled in with any character string. Note the community name that is to be used by BCPii since it will also need to be configured in z/OS.

   Address
   The community name entry used for BCPii should specify an address of **127.0.0.1**.

Network Mask/Prefix
The community name entry used for BCPii should specify a network mask of **255.255.255.255**.

Access Type
Use this field to specify the type of access that is allowed for the community name. The access type for the community name to be used by BCPii *must* be **read/write**.

Asynchronous notification support

The Web Services API asynchronous notification facility provides a means for which client applications can subscribe to and receive notification messages regarding a set of predefined management events. Starting with version 2.16.0 (with the suitable MCL bundle) of the Support Element console the Web Services API asynchronous notification facility has been expanded to support registering for and receiving notifications over the BCPii interface. [Added by feature **bcpii-notifications**]

Note: Please refer to the BCPii documentation for a given OS to determine if client-side support is available.

Notification messages:

Notification messages from BCPii will include the characteristics, properties and bodies outlined in "Notification message formats" on page 80. Each message is represented as a JSON object containing the following fields and values:

Field name	Type	Description
headers	Object	A nested object that contains the following: • The common message properties and characteristics as documented in the "Common message characteristics" on page 80 section. • The extended message properties and characteristics as documented in "Notification message formats" on page 80 for the specific type of notification being received. • The **target-name** message property. The value for this property is the target name of the resource associated with the notification.[1] • The **event-name** message property. The value for this property will be the name of the registered event that triggered the notification.[1]
body	Object	A nested object containing the body of the notification message as documented in "Notification message formats" on page 80 for the specific type of notification being received
[1]BCPii exclusive property		

In addition to the formats documented in "Notification message formats" on page 80 several other formats exist that are exclusive to BCPii. These are as follows:

Console Starting notification:

A Console Starting notification is emitted by the API to report that the console application has started. This will always be the first notification sent once communications are re-established after a reboot cycle.

In addition to the common message properties and characteristics, the following additional message property is provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"console-starting"**.

The body of a Console Starting notification message is a JSON representation of an object that contains the following field and value:

Field name	Type	Description
console-type	String Enum	The type of console: • **"hmc"** - A Hardware Management Console (HMC). • **"se"** - A Support Element console (SE).

Console Heartbeat notification:

A Console Heartbeat notification is emitted by the API to indicate normal operation at regular 2-minute intervals.

In addition to the common message properties and characteristics, the following additional message property is provided for this type of notification:

Message property name	Description
notification-type	Contains the value **"console-heartbeat"**.

The body of a Console Heartbeat notification message is always null.

Operating System Message notification:

The format of an Operating System Message notification will be as described in the "Notification message formats" on page 80 section with one notable exception. The **"message-text"** field of the body's nested os-message-info object may need to be broken up due to size constraints and arrive across multiple messages. If this is the case all other characteristics, properties, and body fields will remain consistent between the messages, and the **"os-messages"** field array will only contain the single os-message-info object. The value of each **"message-text"** can be concatenated in the order of arrival to construct the original message text, split messages will always arrive consecutively.

Notification Registration:

Notification registration is handled by a set of REST operations that are made available exclusively for the BCPii interface. These requests can target the local SE or any other SE that is reachable.

Notification Registration

Notification registration is handled by a set of REST operations that are made available exclusively for the BCPii interface. These requests can target the local SE or any other SE that is reachable.

Register for Notifications

The `Register for Notifications` operation is used by BCPii to register for notifications. [Added by feature **bcpii-notifications**]

HTTP method and URI

```
POST /api/sessions/operations/register-for-notifications
```

Request body contents

The request body is expected to contain a JSON object with the following fields:

Field name	Type	Rqd/ Opt	Description
`event-names`	Array of String Enum	Required	The names of the events to register for. The values here can be matched to the **event-name** notification message property to determine what type of event is being received. The following are the set of valid event types: • **"status-change"** – include Status Change events. An optional **status-change-filter** field can be included to limit what status events are emitted. • **"property-change"** – include Property Change events. An optional **property-change-filter** field can be included to limit what property events are emitted. • **"inventory-change"** – include Inventory Change events. An optional **inventory-change-filter** field can be included to limit what inventory events are emitted. • **"job-completion"** – include completion events of jobs created by this source partition. • **"audit-log-entry"** – include events of new entries to the console's audit log. • **"security-log-entry"** – include events of new entries to the console's security log. • **"console-event-log-entry"** – include events of new entries to the console's event log. • **"os-message"** – include events of new OS messages on registered objects. When this event is included, specifying for which objects is required with the **os-messages-filter** field. • **"console-starting"** - include Console Starting events. • **"console-shutting-down"** - include Console Shutting Down events. • **"disabled-wait"** - include Disabled Wait events. An optional **disabled-wait-filter** field can be included to limit what disabled wait events are emitted. • **"capacity-change"** - include Capacity Change events. • **"capacity-record-change"** - include Capacity Record Change events.
`status-change-filter`	object-change-filter object	Optional	The object-change-filter object, as described in Table 59 on page 136 is used to specify the objects and/or elements on which to register for Status Change events. If omitted, registration will be on all objects and elements that support Status Change events. It is ignored if **status-change** is not included in **event-names**.

Field name	Type	Rqd/ Opt	Description
`property-change-filter`	Array of property-change-filter objects	Optional	An array of property-change-filter objects, as described in Table 60 on page 137 is used to specify the properties, objects and/or elements on which to register for Property Change events. Providing multiple filter objects allows registering for type-specific properties on different types of objects. An event need only pass one filter to be sent. The minimum array size is 1. If omitted, registration will be on all properties of all objects and elements that support Property Change events. It is ignored if **property-change** is not included in **event-names**.
`inventory-change-filter`	object-change-filter object	Optional	The object-change-filter object, as described in Table 59 on page 136 is used to specify the objects and/or elements on which to register for Inventory Change events. If omitted, registration will be on all objects and elements that support Inventory Change events. It is ignored if **inventory-change** is not included in **event-names**.
`os-message-filter`	os-message-filter object	Optional*	The os-message-filter object, as described in Table 61 on page 137 is used to specify the objects on which to register for OS message events. * - This is required if **os-message** is included in **event-names**. Otherwise, it is ignored.
`disabled-wait-filter`	disabled-wait-filter object	Optional	The disabled-wait-filter object, as described in Table 548 on page 1431 is used to specify the objects and/or elements on which to register for Disabled Wait events. If omitted, registration will be on all objects and elements that support Disabled Wait events. It is ignored if **disabled-wait** is not included in **event-names**.

Each nested disabled-wait-filter object contains the following fields:

Table 548. disabled-wait-filter nested object

Field name	Type	Description
objects	Array of String/ Object URI Pattern	Allows matching on any number of objects by their **object-uri** property. Each value must be in the form of an Object URI Pattern except the **object-classification** must be specified and describe objects that support disabled wait messages. These are objects with a classification of **"logical-partitions"**. The minimum array size is 1

Response body contents

On successful completion the response is a JSON object with the following field:

Field name	Type	Description
`registration-id`	String (36)	The unique identifier for the created registration. The **registration-id** is in the form of a UUID.

Description

This operation creates a notification registration. Once registered the client will begin to receive notification messages pertaining to the registered events which pass the provided filters. The **registration-id** returned in the response body can be provided on the `Update Notifications Registration` and `Delete Notifications Registration` operations should modifications to the registration be necessary.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned and the response body is provided as described in "Response body contents" on page 1432.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	377	The local system does not support BCPii v2 asynchronous notification routing.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Update Notifications Registration

The `Update Notifications Registration` operation is used by BCPii to update an existing registration. [Added by feature **bcpii-notifications**]

HTTP method and URI

```
POST /api/sessions/operations/update-notifications-registration
```

Request body contents

The request body is expected to contain a JSON object identical to the `Register for Notifications` operation's Request body contents with the following additional field:

Field name	Type	Rqd/ Opt	Description
`registration-id`	String (36)	Required	The unique identifier for the registration to be updated.

Description

This operation updates an existing registration to the values in the request body.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	7	The provided **registration-id** does not designate a known registration for the session.

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Delete Notifications Registration

The `Delete Notifications Registration` operation is used by BCPii to delete an existing registration. [Added by feature **bcpii-notifications**]

HTTP method and URI

```
POST /api/sessions/operations/delete-notifications-registration
```

Request body contents

The request body is expected to contain a JSON object with the following field:

Field name	Type	Rqd/ Opt	Description
`registration-id`	String (36)	Required	The unique identifier for the registration to be deleted.

Description

This operation deletes an existing registration.

HTTP status and reason codes

On success, HTTP status code 204 (No Content) is returned.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See "Common request validation reason codes" on page 66 for a list of the possible reason codes.
404 (Not Found)	7	The provided **registration-id** does not designate a known registration for the session..

Additional standard status and reason codes can be returned, as described in Chapter 3, "Invoking API operations," on page 59.

Get Notifications Registrations

The `Get Notifications Registrations` operation is used by BCPii to query existing registrations. [Added by feature **bcpii-notifications**]

HTTP method and URI

```
GET /api/sessions/operations/get-notifications-registrations
```

Query parameters:

Field name	Type	Description
`registration-id`	String	A filter query parameter used to limit the returned objects to those that have a matching **registration-id**. This parameter can be specified multiple times.

Response body contents

On successful completion the response is a JSON object with the following field.

Field name	Type	Description
`registrations`	Array of objects	Array of nested objects as described below.

Each nested object is identical to the Register for Notifications operation's Request body contents, with all fields specified and the following additional field:

Field name	Type	Rqd/ Opt	Description
`registration-id`	String (36)	Required	The unique identifier of the registration.

Description

This operation queries the existing registrations for a session. In the case where no registrations exist or pass the query filter an empty array is provided and the operation completes successfully.

HTTP status and reason codes

On success, HTTP status code 200 (OK) is returned.

The following HTTP status codes are returned for the indicated errors. The response body is a standard error response body providing the reason code indicated and an associated error message.

HTTP error status code	Reason code	Description
400 (Bad Request)	Various	Errors were detected during common request validation. See “Common request validation reason codes” on page 66 for a list of the possible reason codes.

Additional standard status and reason codes can be returned, as described in Chapter 3, “Invoking API operations,” on page 59.

Security

The BCPii security controls will be checked before sending a notification. The partition must be enabled to receive commands from the object associated with a notification. Apart from the Console Heartbeat, which is sent regardless, if receive permissions are not enabled the notification will be implicitly filtered

and not sent. Please refer to “Security controls” on page 1425 for details on enabling these permissions. [Added by feature **bcpii-notifications**]

Appendix B. Enum values for a type of managed objects within User Roles

The valid Enum values that can be used to specify a class of managed objects within User Role objects are listed in the following table:

Table 549. Enum values for a class of managed objects

Enum for type name	Type
accelerator-adapter	zEnterprise Data Compression (zEDC) adapter
chpid	Channel path identifier used to map to a specific channel and device number
cloud-network-adapter	Cloud Network adapter
cp	Physical processor in the CPC
cpc	CPC defined to the console through Add Object Definition
crypto	Physical Crypto adapter providing support for cryptographic operations
crypto-adapter	Cryptographic coprocessor adapter
fc-storage-group	Fibre Connection (FICON) storage group
fcp-storage-group	FCP storage group
fcp-tape-link	FCP tape link
ficon-adapter	Fibre Connection (FICON) adapter
fid	Logical identifier associated with virtual function on a PCIe native adapter
hipersockets-adapter	Memory-based network adapter
ism-adapter	Internal Shared Memory Communication (SMC-D) adapter
lpar-image	Partition where an operating system is run
nvme-adapter	Non-volatile Memory express (NVMe) adapter
nvme-storage-group	Non-volatile Memory express (NVMe) storage group
osa-adapter	Open Systems Adapter (OSA) network adapter
partition	Dynamic Partition Manager (DPM) partition
PartitionLink	Partition Link
pattern-match-group	Group containing objects whose name matches a specified pattern
pchid	Physical channel identifier associated with channel hardware (excludes Cryptos)
roce-adapter	RDMA over Converged Ethernet adapter
secure-boot-certificate	Secure Boot Certificate [Added by feature **secure-boot-with-certificates**]

© Copyright IBM Corp. 2022, 2024

Table 549. Enum values for a class of managed objects (continued)	
Enum for type name	**Type**
storage-template	Storage resource definition template
system-manual-definition	Template used to manually define a system object to the Hardware Management Console
tape-library	Tape library
undefined-cpc	CPC that is discovered in the console domain but not defined to the console
user-defined-group	A group of objects defined by a user for organizing objects with a similar purpose or to assign different activation profiles

Appendix C. Enum values for the User Role object

The valid Enum values for the **name** property of User Role objects with a **type** of **"system-defined"** are listed in the following table:

*Table 550. Enum values for the **name** property of User Role objects with a **type** of **"system-defined"***

Enum for system-defined User Role name	System-defined User Role	Valid as an associated User Role
hmc-access-administrator-tasks	Access Administrator Tasks	Yes
hmc-advanced-operator-tasks	Advanced Operator Tasks	Yes
hmc-all-resources	All Resources	No
hmc-all-system-managed-objects	All System Managed Objects	No
hmc-defined-system-managed-objects	Defined System Managed Objects	No
hmc-energy-administrator-tasks	Energy Administrator Tasks	Yes
hmc-operator-tasks	Operator Tasks	Yes
hmc-service-representative-tasks	Service Representative Tasks	Yes
hmc-storage-administrator-objects	Storage Administrator Objects	No
hmc-storage-administrator-tasks	Storage Administrator Tasks	Yes
hmc-system-programmer-tasks	System Programmer Tasks	Yes

© Copyright IBM Corp. 2022, 2024

Appendix D. Enum values for the Task object

The valid Enum values for the **name** property of Task objects are listed in the following table:

*Table 551. Enum values for the **name** property of Task objects*

Enum for task name	Task	view-only-mode supported
access-removable-media	Access Removable Media	false
activate-base	Activate	false
adapter-details	Adapter Details	false
add-object-definition	Add Object Definition	false
add-systems-to-ctn	Add Systems to CTN	false
adjust-leap-second-offset	Adjust Leap Second Offset	false
adjust-time	Adjust Time	false
adjust-time-zone-offset	Adjust Time Zone Offset	false
alternate-support-element	Alternate Support Element	false
alternate-support-element-engineering-changes	Alternate Support Element Engineering Changes (ECs)	false
analyze-console-internal-code	Analyze Console Internal Code	false
archive-security-logs-base	Archive Security Logs	false
archive-security-logs-z	Archive Security Logs	false
audit-and-log-management	Audit and Log Management	false
authorize-internal-code-changes	Authorize Internal Code Changes	false
automatic-activation	Automatic Activation	false
backup-critical-console-data	Backup Critical Console Data	false
backup-critical-data	Backup Critical Data	false
block-automatic-licensed-internal-code-change-installation	Block Automatic Licensed Internal Code Change Installation	false
cancel-scheduled-update	Cancel Scheduled Update	false
certificate-details	Certificate Details [Added by feature **secure-boot-with-certificates**]	false
certificate-management	Certificate Management	false
change-console-internal-code	Change Console Internal Code	false
change-internal-code	Change Internal Code	false
change-lpar-controls	Change LPAR Controls	true
change-lpar-group-controls	Change LPAR Group Controls	true
change-lpar-i-o-priority-queuing	Change LPAR I/O Priority Queuing	false
change-lpar-security	Change LPAR Security	false

© Copyright IBM Corp. 2022, 2024

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
change-object-definition	Change Object Definition	false
change-object-options	Change Object Options	false
concurrent-upgrade-engineering-changes	Concurrent Upgrade Engineering Changes (ECs)	false
configure-3270-emulators	Configure 3270 Emulators	false
configure-backup-settings	Configure Backup Settings	false
configure-channel-path-on-off	Configure Channel Path On/Off	true
configure-data-replication	Configure Data Replication	false
configure-external-time-source	Configure External Time Source	false
configure-storage-storageadmin	Configure Storage – Storage Administrator	false
configure-storage-sysprog	Configure Storage - System Programmer	false
configure-system-policies	Configure System Policies	false
connect-systems-to-key-managers	Connect Systems to Key Managers	false
console-default-user-settings	Console Default User Settings	false
console-messenger	Console Messenger	false
control-pulse-per-second-signal	Control Pulse Per Second Signal	false
copy-console-logs-to-media	Copy Console Logs to Media	false
create-certificate-signing-request	Create Certificate Signing Request	false
create-hipersockets-adapter	Create HiperSockets Adapter	false
create-partition-link	Create Partition Link	false
create-self-signed-certificate	Create Self-Signed Certificate	false
create-welcome-text	Create Welcome Text	false
customer-information	Customer Information	false
customize-activity-profiles	Customize Activity Profiles	false
customize-api-settings	Customize API Settings	false
customize-automatic-logon	Customize Automatic Logon	false
customize-console-date-time	Customize Console Date/Time	false
customize-console-services	Customize Console Services	false
customize-customer-information	Customize Customer Information	false
customize-delete-activation-profiles	Customize/Delete Activation Profiles	false
customize-network-settings-base	Customize Network Settings	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
customize-outbound-connectivity	Customize Outbound Connectivity	false
customize-product-engineering-access	Customize Product Engineering Access	false
customize-remote-service	Customize Remote Service	false
customize-scheduled-operations	Customize Scheduled Operations	false
customize-scheduled-operations-c	Customize Scheduled Operations	false
customize-support-element-date-time	Customize Support Element Date/Time	false
deactivate-base	Deactivate	false
deconfigure-ctn	Deconfigure CTN	false
delete-hipersockets-adapter	Delete HiperSockets Adapter	false
delete-partition	Delete Partition	false
delete-partition-link	Delete Partition Link	false
domain-security	Domain Security	false
dump-partition	Dump	false
edit-certificates	Edit Certificates	false
edit-frame-layout	Edit Frame Layout	false
enable-dynamic-partition-manager	Enable Dynamic Partition Manager	false
enable-ftp-access-to-mass-storage-media	Enable FTP Access to Mass Storage Media	false
enable-i-o-priority-queuing	Enable I/O Priority Queuing	false
energy-optimization-advisor	Energy Optimization Advisor	false
engineering-changes	Engineering Changes (ECs)	false
environmental-efficiency-statistics	Environmental Dashboard [Added by feature **environmental-metrics**]	false
export-certificates-directly-to-key-managers	Export Certificates Directly to Key Managers	false
export-certificates-to-email	Export to Email	false
export-certificates-to-filesystem	Export to File System	false
export-certificates-to-ftp	Export to FTP	false
export-certificates-to-key-managers	Export Certificates to Key Managers	false
export-certificates-to-usb	Export to USB	false
export-ctn-data	Export CTN data (.xls)	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
export-import-profile-data	Export/Import Profile Data (API only)	false
export-wwpns	Export WWPNs	false
fibre-channel-analyzer	Fibre Channel Analyzer	false
file-dialog	File Dialog	false
firmware-details	Firmware Details	false
format-media	Format Media	false
generate-token	Generate Token	false
getting-started-with-dynamic-partition-manager	Getting Started with Dynamic Partition Manager	false
grouping	Grouping	false
hardware-messages	Hardware Messages	true
idaa-callhome-config	Configure IDAA Call-Home	false
image-details	Image Details	false
import-access-control-file	Manage Product Engineering Access Control File	false
import-key-manager-certificate	Import Key Manager Certificate	false
import-signed-certificate	Import Signed Certificate	false
input-output-configuration-save-and-restore	Input/Output (I/O) Configuration Save and Restore	false
installation-complete-report	Installation Complete Report	false
integrated-3270-console	Integrated 3270 Console	false
integrated-ascii-console	Integrated ASCII Console	false
join-existing-ctn	Join Existing CTN	false
lic-security-mode	Change Licensed Internal Code Security Mode	false
limitedlic-security-mode	Change Limited Licensed Internal Code Security Mode (SE task)	false
load	Load	false
load-from-removable-media-or-server	Load from Removable Media or Server	false
logical-processor-add	Logical Processor Add	false
manage-adapters	Manager Adapters	false
manage-coupling-facility-port-enablement	Manage Coupling Facility Port Enablement	true
manage-flash-allocation	Manage Flash allocation	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
manage-key-manager-connections	Manage Key Manager Connections	false
manage-ldap-server-definitions	Manage LDAP Server Definitions	false
manage-password-rules	Manage Password Rules	false
manage-power-service-state	Manage Power Service State	false
manage-print-screen-files	Manage Print Screen Files	false
manage-processor-sharing	Manage Processor Sharing	false
manage-remote-connections	Manage Remote Connections	false
manage-remote-firmware-updates	Manage Remote Firmware Updates	false
manage-remote-support-requests	Manage Remote Support Requests	false
manage-ssh-keys	Manage SSH Keys	false
manage-syslog-servers	Manage Syslog Servers	false
manage-system-time	Manage System Time	false
manage-user-patterns	Manage User Patterns	false
manage-user-roles	Manage User Roles	false
manage-user-templates	Manage User Templates	false
manage-users	Manage Users	false
manage-web-services-api-logs	Manage Web Services API Logs	false
mobile-app-preferences	HMC Mobile Settings	false
modify-assigned-server-roles	Modify Assigned Server Roles	false
monitor-system-events	Monitor System Events	false
monitors-dashboard	Monitors Dashboard	false
multi-factor-auth	Manage Multi-factor Authentication	false
network-diagnostic-information	Network Diagnostic Information	false
network-diagnostic-information-test-se-communication	Test Support Element Communication	false
new-partition	New Partition	false
object-locking-settings	Object Locking Settings	false
offload-virtual-retain-data-to-removable-media	Offload Problem Analysis Data to Removable Media	false
operating-system-messages	Operating System Messages	true
osa-advanced-facilities	OSA Advanced Facilities	true
partition-details	Partition Details	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
partition-details-controls	Partition Details - Controls	false
partition-link-details	Partition Link Details	false
pchid-details	Adapter Details	false
perform-console-repair-action	Perform a Console Repair Action	false
perform-model-conversion	Perform Model Conversion	false
perform-problem-analysis	Perform Problem Analysis	false
perform-transfer-rate-test	Perform Transfer Rate Test	false
product-engineering-directed-changes	Product Support Directed Changes	false
psw-restart	PSW Restart	false
reassign-channel-path-ids	Reassign Channel Path IDs	false
reassign-devices	Reassign Devices	false
reassign-hardware-management-console	Reassign Hardware Management Console	false
reassign-i-o-path	Reassign I/O Path	false
reboot-support-element	Reboot Support Element	false
rebuild-vital-product-data	Rebuild Vital Product Data	false
remote-hardware-management-console	Remote Hardware Management Console	false
remote-service	Remote Service	false
remove-key-manager-connections	Remove Key Manager Connections	false
remove-object-definition	Remove Object Definition	false
remove-systems-from-ctn	Remove Systems from CTN	false
rename-ctn	Rename CTN	false
report-problem-base	Report a Problem	false
report-problem-z	Report a Problem	false
reset-clear	Reset Clear	false
reset-normal	Reset Normal	false
restore-config	Import Dynamic Partition Manager Configuration	false
retrieve-backup-file-from-ftp	Retrieve Backup or Upgrade Data	false
retrieve-internal-code	Retrieve Internal Code	false
save-restore-customizable-console-data	Save/Restore Customizable Console Data	false
save-stp-debug-data	Save STP Debug Data	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
save-upgrade-data	Save Upgrade Data	false
save-upgrade-ftp	Save Upgrade Data	false
se-advanced-facilities	Advanced Facilities	true
se-change-lpar-cryptographic-controls	Change LPAR Cryptographic Controls	false
se-change-mirror-time	Change Mirror Time (SE task)	false
se-channel-interface-tests	Channel Interface Tests	false
se-channel-pchid-assignment	Channel PCHID Assignment	false
se-channel-problem-determination	Channel Problem Determination	false
se-check-internal-code-change-dependencies	Check Dependencies (SE task)	false
se-checkout-tests	Checkout Tests (SE task)	false
se-chpid-details	CHPID Details (SE task)	false
se-cleanup-discontinuance	Cleanup Discontinuance (SE task)	false
se-configure-on-off	Configure On/Off	true
se-cp-details	CP Details (SE task)	false
se-crypto-details	Crypto Details (SE task)	false
se-cryptographic-configuration	Cryptographic Configuration	true
se-cryptographic-management	Cryptographic Management	true
se-define-clonable-internal-code-levels	Define Clonable Internal Code Levels (SE task)	false
se-delete-lpar-dump-data	Delete LPAR Dump Data (SE task)	false
se-display-adapter-id	Display Adapter ID	false
se-display-or-alter	Display or Alter (SE task)	false
se-dump-lpar-data	Dump LPAR Data (SE task)	false
se-dump-machine-loader-data	Dump Machine Loader Data (SE task)	false
se-edit-lpar-internal-code-change	Edit LPAR Internal Code Change (SE task)	false
se-enable-disable-dynamic-channel-subsystem	Enable/Disable Dynamic Channel Subsystem	false
se-energy-optimization-advisor	Energy Optimization Advisor (SE task)	false
se-fcp-configuration	FCP Configuration	false
se-fcp-npiv-mode-on-off	FCP NPIV Mode On/Off	false
se-fid-details	FID Details (SE task)	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
se-force-channel-internal-code-change	Force Channel Internal Code Change (SE task)	false
se-ifcc-and-other-errors	IFCC and Other Errors (SE task)	false
se-input-output-configuration	Input/Output (I/O) Configuration	false
se-interrupt	Interrupt (SE task)	false
se-load-from-removable-media-or-server	Load from Removable Media or Server (SE task)	false
se-load-processor-from-file	Load Processor From File (SE task)	false
se-lpar-internal-code-change-utility	LPAR Internal Code Change Utility (SE task)	false
se-manage-pci-system-services	Manage PCI System Services	false
se-migrate-channel-configuration-files	Migrate Channel Configuration Files (SE task)	false
se-msq-processor-test	MSQ Processor Test (SE task)	false
se-network-traffic-analyzer-authorization	Network Traffic Analyzer Authorization	false
se-nnhb-monitor	NNhb Monitor (SE task)	false
se-non-disruptive-hardware-change	Nondisruptive Hardware Change (SE task)	false
se-offload-pa-data-to-hmc-removable-media	Offload Problem Analysis Data to HMC removable Media (SE task)	false
se-perform-model-conversion	Perform Model Conversion	false
se-power-off	Power Off (SE task)	false
se-power-on	Power On (SE task)	false
se-power-on-reset	Power-on Reset (SE task)	false
se-prepare-system-for-discontinuance	Prepare System For Discontinuance (SE task)	false
se-query-channel-crypto-configure-on-off-pending	Query Channel/Crypto Configure On/Off Pending	false
se-query-coupling-facility-reactivations	Query Coupling Facility Reactivations	false
se-query-internal-code-changes-pending-power-on-reset	Query Internal Code Changes Pending Power-on Reset (SE task)	false
se-reassign-io-path	Reassign I/O Path (SE task)	false
se-redundant-io-interconnect-status-and-control	Redundant I/O Interconnect Status and Control	false
se-release-io-path	Release I/O Path	false
se-reset-error-thresholds	Reset Error Thresholds	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
se-reset-swap-channel-path	Reset Swap Channel Path (SE task)	false
se-selective-channel-patch-controls	Selective Channel Patch Controls (SE task)	false
se-service-on-off	Service On/Off	false
se-service-required-state-query	Service Required State Query	false
se-show-led	Show LED	false
se-start-processor	Start Processor (SE task)	false
se-stop-processor	Stop Processor (SE task)	false
se-stop-processor-on-cp-address-match	Stop Processor on CP Address Match (SE task)	false
se-storage-information	Storage Information	false
se-store-status	Store Status (SE task)	false
se-swap-channel-patch	Swap Channel Path (SE task)	false
se-transmit-vital-product-data	Transmit Vital Product Data (SE task)	false
se-update-io-world-wide-port-number	Update I/O World Wide Port Number (SE task)	false
se-update-pci-adapter-internal-code	Update PCI Adapter Internal Code	false
se-view-hardware-configuration	View Hardware Configuration (SE task)	false
se-view-internal-code-changes-summary	View Internal Code Changes Summary	false
se-view-lpar-cryptographic-controls	View LPAR Cryptographic Controls	false
secure-boot-certificate-management	Secure Boot Certificate Management [Added by feature **secure-boot-with-certificates**]	false
secure-boot-certificate-management-assign-certificate	Assign Secure Boot Certificates [Added by feature **secure-boot-with-certificates**]	false
secure-boot-certificate-management-import-certificate	Import Secure Boot Certificates [Added by feature **secure-boot-with-certificates**]	false
secure-execution-clear-keys	Delete Secondary Secure Execution Key	false
secure-execution-import-keys	Import Secure Execution Keys (SE task)	false
secure-execution-manage-keys	Manage Secure Execution Keys	false
service-status	Service Status	false
set-ctn-member-restriction	Set CTN Member Restriction	false
set-power-cap	Set Power Cap	false
set-power-saving	Set Power Saving	false
set-time-server-power-failover	Set Time Server Power Failover	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
setup-new-ctn	Setup New CTN	false
shutdown-or-restart	Power Off or Restart	false
single-object-operations	Single Object Operations	false
single-step-console-internal-code	Single Step Console Internal Code	false
single-step-internal-code-changes	Single Step Internal Code Changes	false
split-to-new-ctn	Split to New CTN	false
start-all	Start All Processors	false
start-partitions	Start	false
start-systems	Start	false
stop-all	Stop All Processors	false
stop-partitions	Stop	false
stop-systems	Stop	false
system-activity-display	Activity	false
system-details	System Details	false
system-information	System Information	false
system-input-output-configuration-analyzer	System Input/Output Configuration Analyzer	false
tip-of-the-day	Tip of the Day	false
toggle-lock	Toggle Lock	false
transmit-console-service-data	Transmit Console Service Data	false
transmit-service-data	Transmit Service Data	false
transmit-vital-product-data-base	Transmit Vital Product Data	false
transmit-vital-product-data-z	Transmit Vital Product Data	false
user-settings	User Settings	false
users-and-tasks	Users and Tasks	false
view-activation-profiles	View Activation Profiles	false
view-adapter-security	View Adapter Security	false
view-certificate-details	View Certificate Details	false
view-console-events	View Console Events	false
view-console-information	View Console Information	false
view-console-service-history	View Console Service History	false
view-console-tasks-performed	View Console Tasks Performed	false
view-external-time-source	View External Time Source	false

*Table 551. Enum values for the **name** property of Task objects (continued)*		
Enum for task name	**Task**	**view-only-mode supported**
view-frame-layout	View Frame Layout	false
view-licenses	View Licenses	false
view-partition-resource-assignments	View Partition Resource Assignments	false
view-pulse-per-second-signal	View Pulse Per Second Signal	false
view-security-logs	View Security Logs	false
view-service-history	View Service History	false
virtual-media-load	Manage Console Recovery	false
virtual-se-management	Virtual Support Element Management	false
whats-new	What's New	false
zeroize-crypto-domain	Zeroize Crypto Domain (API only)	false

Appendix E. Notices

This information was developed for products and services offered in the US.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprise is entirely coincidental.

Trademarks

IBM, the IBM logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corporation, in the United States and/or other countries. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on http://www.ibm.com/trademark.

© Copyright IBM Corp. 2022, 2024

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Unix is a trademark of The Open Group in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.

Class A Notices

The following Class A statements apply to this IBM product. The statement for other IBM products intended for use with this product will appear in their accompanying manuals.

Federal Communications Commission (FCC) Statement

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user will be required to correct the interference at his own expense.

Properly shielded and grounded cables and connectors must be used in order to meet FCC emission limits. IBM is not responsible for any radio or television interference caused by using other than recommended cables and connectors or by unauthorized changes or modifications to this equipment. Unauthorized changes or modifications could void the user's authority to operate the equipment.

This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Industry Canada Compliance Statement

This Class A digital apparatus complies with Canadian ICES-003.

United Kingdom Notice

This product may cause interference if used in residential areas. Such use must be avoided unless the user takes special measures to reduce electromagnetic emissions to prevent interference to the reception of radio and television broadcasts.

Avis de conformité à la réglementation d'Industrie Canada

Cet appareil numérique de la classe A est conforme à la norme NMB-003 du Canada.

European Community Compliance Statement

This product is in conformity with the protection requirements of EU Council Directive 2014/30/EU on the approximation of the laws of the Member States relating to electromagnetic compatibility. IBM cannot accept responsibility for any failure to satisfy the protection requirements resulting from a non-recommended modification of the product, including the fitting of non-IBM option cards.

This product has been tested and found to comply with the limits for Class A Information Technology Equipment according to European Standard EN 55032. The limits for Class A equipment were derived

for commercial and industrial environments to provide reasonable protection against interference with licensed communication equipment.

European Community contact:
IBM Deutschland GmbH
Technical Regulations, Department M372
IBM-Allee 1, 71139 Ehningen, Germany
Tele: +49 (0) 800 225 5423 or +49 (0) 180 331 3233
email: halloibm@de.ibm.com

Warning: This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user may be required to take adequate measures.

Japan Voluntary Control Council for Interference (VCCI) Notice

この装置は、クラス A 機器です。 この装置を住宅環境で使用すると電波妨害を引き起こすことがあります。 この場合には使用者が適切な対策を講ずるよう要求される ことがあります。 ＶＣＣＩ－Ａ

The following is a summary of the Japanese VCCI statement above:

This is a Class A product based on the standard of the VCCI Council. If this equipment is used in a domestic environment, radio interference may occur, in which case the user may be required to take corrective actions.

Japan Electronics and Information Technology Industries Association (JEITA) Notice

（一社）電子情報技術産業協会　高調波電流抑制対策実施
要領に基づく定格入力電力値：IBM Documentationの各製品
の仕様ページ参照

This statement applies to products less than or equal to 20 A per phase.

高調波電流規格　JIS C 61000-3-2 適合品

These statements apply to products greater than 20 A, single-phase.

高調波電流規格　JIS C 61000-3-2 準用品

本装置は、「高圧又は特別高圧で受電する需要家の高調波抑制対策ガイドライン」対象機器（高調波発生機器）です。
回路分類：６（単相、ＰＦＣ回路付）
換算係数：０

These statements apply to products greater than 20 A per phase, three-phase.

高調波電流規格　JIS C 61000-3-2 準用品

本装置は、「高圧又は特別高圧で受電する需要家の高調波抑制対策ガイドライン」対象機器（高調波発生機器）です。
回路分類　：５（３相、ＰＦＣ回路付）
換算係数　：０

People's Republic of China Notice

警告:在居住环境中,运行此设备可能会造成无线电干扰。

Declaration: This is a Class A product. In a domestic environment, this product may cause radio interference, in which case the user may need to perform practical action.

Taiwan Notice

CNS 13438:

警告使用者：
此為甲類資訊技術設備，
於居住環境中使用時，
可能會造成射頻擾動，在此種情況下，
使用者會被要求採取某些適當的對策。

CNS 15936:

警告：為避免電磁干擾，本產品不應安裝或使用於住宅環境。

IBM Taiwan Contact Information:

台灣IBM 產品服務聯絡方式：
台灣國際商業機器股份有限公司
台北市松仁路7號3樓
電話：0800-016-888

Electromagnetic Interference (EMI) Statement - Korea

이 기기는 업무용(A급)으로 전자파적합등록을 한 기기이오니
판매자 또는 사용자는 이 점을 주의하시기 바라며, 가정외의
지역에서 사용하는 것을 목적으로 합니다.

Germany Compliance Statement

Deutschsprachiger EU Hinweis: Hinweis für Geräte der Klasse A EU-Richtlinie zur Elektromagnetischen Verträglichkeit

Dieses Produkt entspricht den Schutzanforderungen der EU-Richtlinie 2014/30/EU zur Angleichung der Rechtsvorschriften über die elektromagnetische Verträglichkeit in den EU-Mitgliedsstaaten und hält die Grenzwerte der EN 55032 Klasse A ein.

Um dieses sicherzustellen, sind die Geräte wie in den Handbüchern beschrieben zu installieren und zu betreiben. Des Weiteren dürfen auch nur von der IBM empfohlene Kabel angeschlossen werden. IBM übernimmt keine Verantwortung für die Einhaltung der Schutzanforderungen, wenn das Produkt ohne Zustimmung von IBM verändert bzw. wenn Erweiterungskomponenten von Fremdherstellern ohne Empfehlung von IBM gesteckt/eingebaut werden.

EN 55032 Klasse A Geräte müssen mit folgendem Warnhinweis versehen werden:

"Warnung: Dieses ist eine Einrichtung der Klasse A. Diese Einrichtung kann im Wohnbereich Funk-Störungen verursachen; in diesem Fall kann vom Betreiber verlangt werden, angemessene Maßnahmen zu ergreifen und dafür aufzukommen."

Deutschland: Einhaltung des Gesetzes über die elektromagnetische Verträglichkeit von Geräten

Dieses Produkt entspricht dem "Gesetz über die elektromagnetische Verträglichkeit von Geräten (EMVG)". Dies ist die Umsetzung der EU-Richtlinie 2014/30/EU in der Bundesrepublik Deutschland.

Zulassungsbescheinigung laut dem Deutschen Gesetz über die elektromagnetische Verträglichkeit von Geräten (EMVG) (bzw. der EMC EG Richtlinie 2014/30/EU) für Geräte der Klasse A

Dieses Gerät ist berechtigt, in Übereinstimmung mit dem Deutschen EMVG das EG-Konformitätszeichen - CE - zu führen.

Verantwortlich für die Einhaltung der EMV Vorschriften ist der Hersteller:
International Business Machines Corp.
New Orchard Road
Armonk, New York 10504
Tel: 914-499-1900

Der verantwortliche Ansprechpartner des Herstellers in der EU ist:
IBM Deutschland GmbH
Technical Regulations, Abteilung M372
IBM-Allee 1, 71139 Ehningen, Germany
Tel: +49 (0) 800 225 5423 or +49 (0) 180 331 3233
email: halloibm@de.ibm.com

Generelle Informationen:

Das Gerät erfüllt die Schutzanforderungen nach EN 55024 und EN 55032 Klasse A.

Electromagnetic Interference (EMI) Statement - Russia

ВНИМАНИЕ! Настоящее изделие относится к классу А.
В жилых помещениях оно может создавать радиопомехи, для
снижения которых необходимы дополнительные меры

Electromagnetic Interference (EMI) Statement - Kingdom of Saudi Arabia Notice

قد يتسبب هذا المنتج في حدوث تداخل إذا تم استخدامه في المناطق السكنية.

ويجب تجنب هذا الاستخدام ما لم يتخذ المستخدم تدابير خاصة لتقليل الانبعاثات الكهرومغناطيسية لمنع التداخل مع استقبال البث الإذاعي والتلفزيوني.

تحذير: هذا الجهاز متوافق مع الفئة أ من SASO CISPR 32

في البيئة السكنية، قد يتسبب هذا الجهاز في حدوث تداخل لاسلكي.

Index

A

Accept Mismatched Storage Volumes 589
accessibility lxxi
Activate CPC 1054
Activate Logical Partition 1207
Add Adapter Ports 697
Add Candidate Adapter Ports to an FCP Storage Group 571
Add Connection Endpoint 491
Add Member to Custom Group 1005
Add Partition to Capacity Group 429
Add Permission to User Role 927
Add Temporary Capacity 1064
Add User Role to User 907
Add Volume Range 509
API version number, function included in 7
Assign Certificate to Image Activation Profile 1316
Assign Certificate to Logical Partition 1256
Assign Certificate to Partition 353
assistive technologies lxxi
Attach Storage Group to Partition 266
Attach Tape Link to Partition 343
Authorize Remote Firmware Updates 870

B

Bace Control Program internal interface 1411
BCPii 1411

C

Cancel Job 156
Change Adapter Type 401
Change Crypto Domain Configuration 308
Change Crypto Type 386
Change Logon Password 126
Change STP-only Coordinated Timing Network 1074
Console Delete Retrieved Internal Code 884
Console Single Step Install 876
CPC Delete Retrieved Internal Code 1162
CPC Install and Activate 1157
CPC Single Step Install 1136
Create Capacity Group 423
Create Custom Group 1002
Create Group Profile 1370
Create HBA 321
Create Hipersocket 388
Create Image Activation Profile 1321
Create LDAP Server Definition 982
Create Load Activation Profile 1353
Create Metrics Context 169
Create MFA Server Definition 993
Create NIC 291
Create Partition 239
Create Partition Link 742
Create Password Rule 967
Create Reset Activation Profile 1285
Create Server-Sent Events Stream 134
Create Storage Fabric 451
Create Storage Group 547
Create Storage Path 513
Create Storage Site 439
Create Storage Template 644
Create Tape Link 685
Create User 911
Create User Pattern 952
Create User Role 933
Create Virtual Function 283

D

Deactivate CPC 1057
Deactivate Logical Partition 1212
Decline Console Service 848
Decline CPC Service 1100
Decrease Crypto Configuration 310
Define Storage Control Unit 502
Define Storage Subsystem 482
Define Storage Switch 466
Delete Capacity Group 426
Delete Certificate 1267
Delete Completed Job Status 154
Delete Console Hardware Message 843
Delete Console Remote Firmware Update 868
Delete CPC Hardware Message 1093
Delete CPC Remote Firmware Update 1119
Delete Custom Group 1004
Delete Group Profile 1373
Delete HBA 324
Delete Hipersocket 390
Delete Image Activation Profile 1338
Delete LDAP Server Definition 984
Delete Load Activation Profile 1360
Delete Logical Partition OS Message 1243
Delete Metrics Context 175
Delete MFA Server Definition 995
Delete NIC 296
Delete Notifications Registration 1433
Delete Partition 245
Delete Partition Asynchronously 247
Delete Partition Link 755
Delete Partition OS Message 339
Delete Password Rule 969
Delete Reset Activation Profile 1289
Delete Secure Execution Key 1145
Delete Server-Sent Events Stream 140
Delete Storage Fabric 454
Delete Storage Group 553
Delete Storage Path 516
Delete Storage Site 442
Delete Storage Template 647
Delete Tape Link 695
Delete User 915
Delete User Pattern 955

Delete User Role 935
Delete Virtual Function 286
Detach Storage Group from Partition 319
Detach Tape Link from Partition 346
Device number constraints 197
Discover Tape Libraries 672
Dump Partition 271

E

Enum values for a class of managed objects 1437
Enum values for Task objects 1441
Enum values for User Role objects 1439
Establish Shared Secret Key 121
Export Profiles 1060
Export WWPN List 1101

F

FICON storage configuration 195
Flash memory adapters 187
Fulfill FCP Storage Volume 587
Fulfill FICON Storage Volume 580
Fulfill FICON Storage Volumes 583

G

Get Adapter Properties 381
Get ASCII Console WebSocket URI 341
Get Capacity Group Properties 427
Get Capacity Record Properties 1380
Get Certificate Properties 1268
Get Connected VNICs of a Virtual Switch 416
Get Connection Report 608
Get Console Audit Log 824
Get Console Events Log 833
Get Console Hardware Message Properties 841
Get Console Notification Preferences for Device 873
Get Console Properties 812
Get Console Remote Firmware Update Properties 867
Get Console Security Log 830
Get Console Service Request Information 846
Get CPC Audit Log 1078
Get CPC Energy Management Data 1397
Get CPC Events Log 1084
Get CPC Hardware Message Properties 1091
Get CPC Historical Sustainability Data 1152
Get CPC Notification Preferences for Device 856
Get CPC Properties 1039
Get CPC Remote Firmware Update Properties 1117
Get CPC Security Log 1081
Get CPC Service Request Information 1097
Get Custom Group Properties 1000
Get Encoded Certificate 1271
Get Energy Optimization Advice Details 1402
Get Energy Optimization Advice Summary 1399
Get Group Profile Properties 1366
Get HBA Properties 328
Get Image Activation Profile Properties 1310
Get Inventory 160
Get LDAP Server Definition Properties 978
Get Load Activation Profile Properties 1349
Get Logical Partition Historical Sustainability Data 1263
Get Logical Partition Properties 1200
Get Logical Partition Resource Assignments 1121
Get LPAR Controls 1122
Get Metrics 172
Get MFA Server Definition Properties 990
Get Mobile App Preferences 852
Get Network Port Properties 394
Get NIC Properties 298
Get Notification Topics 131
Get Notifications Registrations 1434
Get Partition Historical Sustainability Data 350
Get Partition Link Properties 759
Get Partition Properties 250
Get Partitions Assigned to Adapter 392
Get Partitions for a Storage Group 601
Get Partitions for a Tape Link 713
Get Password Rule Properties 963
Get Reset Activation Profile Properties 1281
Get Server-Sent Events Stream Last Event ID 143
Get Storage Control Unit Properties 506
Get Storage Fabric Properties 455
Get Storage Group Histories 617
Get Storage Group Properties 556
Get Storage Path Properties 518
Get Storage Port Properties 398
Get Storage Site Properties 444
Get Storage Subsystem Properties 486
Get Storage Switch Properties 470
Get Storage Template Properties 649
Get Storage Template Volume Properties 659
Get Storage Volume Properties 578
Get Tape Library Properties 666
Get Tape Link Environment Report 729
Get Tape Link Histories 715
Get Tape Link Properties 689
Get Task Properties 940
Get User Pattern Properties 947
Get User Properties 902
Get User Role Properties 923
Get Virtual Function Properties 287
Get Virtual Storage Resource Properties 597
Get Virtual Switch Properties 414
Get Virtual Tape Resource Properties 709

I

IBM z Unified Resource Manager 3
Import CPC Certificate 1147
Import DPM Configuration 1104
Import Profiles 1059
Import Secure Execution Key 1142
Increase Crypto Configuration 305

J

Join STP-only Coordinated Timing Network 1075

K

keyboard
    navigation lxxi

L

Leave STP-only Coordinated Timing Network 1077
List Adapters of a CPC 374
List Capacity Groups of a CPC 421
List Capacity Records 1378
List Certificates 1273
List Console API Features 887
List Console Hardware Messages 839
List CPC API Features 1165
List CPC Hardware Messages 1088
List CPC Objects 1036
List Custom Group Members 1008
List Custom Groups 998
List Group Profiles 1364
List Image Activation Profiles 1307
List LDAP Server Definitions 977
List Load Activation Profiles 1346
List Logical Partitions of CPC 1194
List MFA Server Definitions 988
List OS Messages of a Logical Partition 1239
List OS Messages of a Partition 336
List Partition Links 762
List Partitions of a CPC 234
List Password Rules 961
List Permitted Adapters 377
List Permitted Logical Partitions 1196
List Permitted Partitions 237
List Permitted Virtual Switches 411
List Remote Firmware Updates of a CPC 1115
List Remote Firmware Updates of the Console 864
List Reset Activation Profiles 1279
List Storage Control Units of a Storage Subsystem 500
List Storage Fabrics 449
List Storage Groups 544
List Storage Sites 437
List Storage Subsystems of a Storage Site 480
List Storage Switches of a Storage Fabric 463
List Storage Switches of a Storage Site 461
List Storage Template Volumes of a Storage Template 656
List Storage Templates 642
List Storage Volumes of a Storage Group 575
List Tape Libraries 663
List Tape Links 683
List Tasks 938
List Unmanaged CPCs 850
List User Patterns 946
List User Roles 921
List Users 900
List Virtual Storage Resources of a Storage Group 594
List Virtual Switches of a CPC 409
List Virtual Tape Resources of a Tape Link 707
Load 1218
Load Logical Partition 1224
Load Logical Partition from FTP 1227
Logoff 130
Logon 115

M

managed objects, valid enum values 1437
Modify Partition Link 764
Modify Storage Group Properties 558
Modify Storage Template Properties 650
Modify Tape Link Properties 691
Mount ISO Image 316
Move Storage Subsystem to Storage Site 489
Move Storage Switch to Storage Fabric 475
Move Storage Switch to Storage Site 473

N

navigation
    keyboard lxxi
NVMe Dump 1254
NVMe Load 1251

O

Open OS Message Channel 334, 1237
Open Server-Sent Events Stream 141

P

Perform PSW Restart 281
Provide Requested MFA Information 123
PSW Restart 1230

Q

Query API Version 113
Query Job Status 151

R

Reassign Storage Adapter Port 330
Register for Notifications 1429
Reject Mismatched FCP Storage Volumes 591
Remove Adapter Ports 700
Remove Candidate Adapter Ports from an FCP Storage Group 573
Remove Connection Endpoint 494
Remove Member from Custom Group 1007
Remove Partition from Capacity Group 431
Remove Permission from User Role 930
Remove Temporary Capacity 1067
Remove User Role from User 909
Remove Volume Range 511
Reorder User Patterns 822
Replace Adapter Port 702
Report a Console Problem 882
Report a CPC Problem 1149
Report a Logical Partition Problem 1260
Report a Partition Problem 348
Request Console Service 844
Request CPC Service 1095
Request Tape Library Zoning 670
Resend Request 568, 704
Reset Clear 1216
Reset Normal 1214
Restart Console 819
Revisions lxxi
RoCE adapters 187

S

SCSI Dump 1248
SCSI Load 1245
Send OS Command 332, 1235
Set Auto-Start List 1062
Set CPC Power Capping 1390
Set CPC Power Save 1388
Set Mobile App Preferences 853
Set STP Configuration 1071
Set zCPC Power Capping 1395
Set zCPC Power Save 1393
shortcut keys lxxi
Shutdown Console 821
Start CPC 1049
Start Dump Program 275
Start FCP Storage Discovery 605
Start Logical Partition 1232
Start Partition 262
Stop CPC 1052
Stop Logical Partition 1233
Stop Partition 269
Submit Requests 144
Summary of updates by API version number 7
Swap Current Time Server 1069
Switch Support Elements 1167

T

Task objects, valid enum values 1441

U

Unassign Certificate from Image Activation Profile 1319
Unassign Certificate from Logical Partition 1258
Unassign Certificate from Partition 355
Undefine Storage Control Unit 504
Undefine Storage Subsystem 484
Undefine Storage Switch 468
Undefine Tape Library 665
Unified Resource Manager 3
Unmount ISO Image 317
Update Adapter Firmware 403
Update Adapter Properties 383
Update Capacity Group Properties 433
Update Certificate Properties 1275
Update Console Notification Preferences for Device 875
Update CPC Notification Preferences for Device 860
Update CPC Properties 1048
Update Group Profile Properties 1369
Update HBA Properties 326
Update Image Activation Profile Properties 1315
Update LDAP Server Definition Properties 980
Update Load Activation Profile Properties 1351
Update Logical Partition Properties 1205
Update LPAR Controls 1134
Update MFA Server Definition Properties 991
Update Network Port Properties 396
Update NIC Properties 300
Update Notifications Registration 1432
Update Partition Properties 253
Update Partition Properties Asynchronously 257
Update Password Rule Properties 965
Update Reset Activation Profile Properties 1283
Update Server-Sent Events Stream 139
Update Storage Control Unit Properties 507
Update Storage Fabric Properties 457
Update Storage Path Properties 519
Update Storage Port Properties 399
Update Storage Site Properties 445
Update Storage Subsystem Properties 488
Update Storage Switch Properties 471
Update Tape Library Properties 668
Update Tape Link Environment Report 726
Update User Pattern Properties 949
Update User Properties 904
Update User Role Properties 925
Update Virtual Function Properties 289
Update Virtual Storage Resource Properties 598
Update Virtual Switch Properties 417
Update Virtual Tape Resource Properties 710
Update Welcome Text 871
User Role objects, valid enum values 1439

V

Validate LUN Path 603
Verify Logon Password 129

W

Web Services API 3

Z

Zeroize Crypto Domain 313
zManager 3

SC27-2642-02