

IBM Spectrum Scale
Version 4 Release 2.0

*Administration and Programming
Reference*



IBM Spectrum Scale
Version 4 Release 2.0

*Administration and Programming
Reference*



Note

Before using this information and the product it supports, read the information in “Notices” on page 891.

This edition applies to version 4 release 2 of the following products, and to all subsequent releases and modifications until otherwise indicated in new editions:

- IBM Spectrum Scale ordered through Passport Advantage® (product number 5725-Q01)
- IBM Spectrum Scale ordered through AAS/eConfig (product number 5641-GPF)
- IBM Spectrum Scale for Linux on z Systems (product number 5725-S28)

Significant changes or additions to the text and illustrations are indicated by a vertical line (|) to the left of the change.

IBM welcomes your comments; see the topic “How to send your comments” on page xiv. When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright IBM Corporation 2014, 2016.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Tables	ix
-------------------------	-----------

About this information	xi
---	-----------

Prerequisite and related information	xiii
Conventions used in this information	xiii
How to send your comments	xiv

Summary of changes	xv
-------------------------------------	-----------

Chapter 1. Performing GPFS administration tasks **1**

Requirements for administering a GPFS file system	1
adminMode configuration attribute	2
Common GPFS command principles	3
Specifying nodes as input to GPFS commands	3
Stanza files	4
Listing active GPFS commands	5

Chapter 2. Managing the GPFS cluster **7**

Creating your GPFS cluster	7
Displaying GPFS cluster configuration information	7
Adding nodes to a GPFS cluster	8
Deleting nodes from a GPFS cluster	9
Changing the GPFS cluster configuration data	10
Security mode	20
Running IBM Spectrum Scale without remote root login	21
Configuring sudo	21
Configuring the cluster to use sudo wrapper scripts	22
Configuring a cluster to stop using sudo wrapper scripts	23
Node quorum considerations	23
Node quorum with tiebreaker considerations	23
Displaying and changing the file system manager node	24
Determining how long <code>mmrestripefs</code> takes to complete	24
Starting and stopping GPFS	25

Chapter 3. Managing file systems **27**

Mounting a file system	27
Mounting a file system on multiple nodes	28
GPFS-specific mount options	28
Unmounting a file system	29
Unmounting a file system on multiple nodes	29
Deleting a file system	29
Determining which nodes have a file system mounted	30
Checking and repairing a file system	30
Dynamic validation of descriptors on disk	32
Listing file system attributes	33
Modifying file system attributes	34
Querying and changing file replication attributes	34

Querying file replication	34
Changing file replication attributes	34
Using Direct I/O on a file in a GPFS file system	35
File compression	35
Setting the Quality of Service for I/O operations (QoS)	40
Restripping a GPFS file system	42
Querying file system space	43
Querying and reducing file system fragmentation	44
Querying file system fragmentation	45
Reducing file system fragmentation	45
Protecting data in a file system using backup	46
Protecting data in a file system using the <code>mmbackup</code> command	46
Backing up a file system using the GPFS policy engine	52
Backing up file system configuration information	52
Using APIs to develop backup applications	52
Scale Out Backup and Restore (SOBAR)	53
Scheduling backups using Tivoli Storage Manager scheduler	54
Configuration reference for using Tivoli Storage Manager with IBM Spectrum Scale	54
Options in the Tivoli Storage Manager configuration file <code>dsm.sys</code>	54
Options in the Tivoli Storage Manager configuration file <code>dsm.opt</code>	56
Base Tivoli Storage Manager client configuration files for IBM Spectrum Scale usage	57
Restoring a subset of files or directories from a local file system snapshot	58
Restoring a subset of files or directories from a local fileset snapshot	59
Restoring a subset of files or directories from local snapshots using the sample script	60

Chapter 4. Configuring the CES and protocol configuration **63**

Configuring Cluster Export Services	63
Setting up Cluster Export Services shared root file system	63
Configuring Cluster Export Services nodes	64
Configuring CES protocol service IP addresses	64
Deploying Cluster Export Services packages on IBM Spectrum Scale existing 4.1.1 and above	65
Verifying the final CES configurations	66
Creating and configuring file systems and filesets for exports	66
Managing protocol services	66
Configuring and enabling SMB and NFS protocol services	66
Configuring and enabling the Object protocol service	68
Disabling protocol services	69
Managing protocol user authentication	69

Setting up authentication servers to configure protocol user access	69
Configuring authentication and ID mapping for file access	76
Managing user-defined authentication	89
Configuring authentication for object access	93
Deleting the authentication and the ID mapping configuration	105
Listing the authentication configuration	107
Verifying the authentication services configured in the system	107
Modifying the authentication method	108
Authentication limitations	109
General authentication considerations	112
Configuring with the spectrumscale installation toolkit	112

Chapter 5. Managing protocol data exports 113

Managing SMB shares	113
Creating SMB share	113
Changing SMB share configuration	114
Creating SMB share ACLs	114
Removing SMB shares	114
Listing SMB shares	114
Managing SMB shares using MMC	115
SMB share limitations	119
Managing NFS exports	120
Creating NFS exports	120
Changing NFS export configuration	120
Removing NFS exports	121
Listing NFS exports	121
Multiprotocol exports	121
Multiprotocol export considerations	121

Chapter 6. Managing object storage 123

Understanding and managing Object services	123
Understanding the mapping of OpenStack commands to IBM Spectrum Scale administrator commands	125
Changing Object configuration values	125
Changing the object base configuration to enable S3 API emulation	126
Configuring OpenStack EC2 credentials	128
Managing OpenStack access control lists using S3 API emulation	129
Managing object capabilities	130
Mapping of storage policies to filesets	130
Administering storage policies for object storage	131
Creating storage policy for object compression	132
Adding a region in a multi-region object deployment	132
Administering a multi-region object deployment environment	134
Unified file and object access in IBM Spectrum Scale	135
Identity management modes for unified file and object access	135
Authentication in unified file and object access	140
The objectizer process	140

File path in unified file and object access	141
Administering unified file and object access	142
In-place analytics using unified file and object access	149
Limitations of unified file and object access	150
Constraints applicable to unified file and object access	151
Data ingestion examples	151
curl commands for unified file and object access related user tasks	152
Configuration files for IBM Spectrum Scale for object storage	153

Chapter 7. Managing disks 157

Displaying disks in a GPFS cluster	157
Adding disks to a file system	158
Deleting disks from a file system	158
Replacing disks in a GPFS file system	160
Additional considerations for managing disks	161
Displaying GPFS disk states	162
Disk availability	162
Disk status	162
Changing GPFS disk states and parameters	163
Changing your NSD configuration	165
Changing NSD server usage and failback	166
Enabling and disabling Persistent Reserve	166

Chapter 8. Managing GPFS quotas 169

Enabling and disabling GPFS quota management	169
Default quotas	170
Implications of quotas for different protocols	172
Explicitly establishing and changing quotas	173
Setting quotas for users on a per-project basis	174
Checking quotas	176
Listing quotas	177
Activating quota limit checking	178
Deactivating quota limit checking	178
Changing the scope of quota limit checking	179
Creating file system quota reports	179
Restoring quota files	180

Chapter 9. Managing GUI administrators 183

Chapter 10. Managing GPFS access control lists 187

Traditional GPFS ACL administration	187
Setting traditional GPFS access control lists	188
Displaying traditional GPFS access control lists	189
Applying an existing traditional GPFS access control list	189
Changing traditional GPFS access control lists	190
Deleting traditional GPFS access control lists	190
NFS V4 ACL administration	191
NFS V4 ACL Syntax	191
NFS V4 ACL translation	193
Setting NFS V4 access control lists	194
Displaying NFS V4 access control lists	194
Applying an existing NFS V4 access control list	194
Changing NFS V4 access control lists	194

Deleting NFS V4 access control lists	195
Considerations when using GPFS with NFS V4	
ACLs	195
NFS and GPFS	195
Exporting a GPFS file system using NFS	195
NFS usage of GPFS cache	198
Synchronous writing using NFS	199
Unmounting a file system after NFS export	199
NFS automount considerations	199
Clustered NFS and GPFS on Linux	199
Authorizing protocol users	200
Authorizing file protocol users	200
Authorizing object users.	209
Authorization limitations	214

Chapter 11. GPFS commands 217

gpfs.snap command	221
mmaddcallback command	226
mmadddisk command	239
mmaddnode command	245
mmafmconfig command.	248
mmafmctl command	251
mmafmlocal command	264
mmapplypolicy command	266
mmauth command	276
mmbackup command	281
mmbackupconfig command	290
mmbuildgpl command	292
mmcallhome command	293
mmces command	304
mmcesdr command	313
mmchattr command	321
mmchcluster command	327
mmchconfig command	331
mmchdisk command	354
mmcheckquota command	361
mmchfileset command	365
mmchfs command.	371
mmchlicense command	377
mmchmgr command	379
mmchnode command	381
mmchnodeclass command	385
mmchnsd command	388
mmchpolicy command	391
mmchpool command	394
mmchqos command	396
mmclone command	400
mmcrcluster command	403
mmcrfileset command	408
mmcrfs command	414
mmcrnodeclass command	424
mmcrnsd command	426
mmcrsnapshot command	431
mmdefedquota command	434
mmdefquotaoff command	437
mmdefquotaon command	440
mmdefragfs command	443
mmdelacl command	446
mmdelcallback command	448
mmdeldisk command	449
mmdelfileset command	455

mmdelfs command	458
mmdelnode command	460
mmdelnodeclass command.	463
mmdelnsd command.	465
mmdelnsnapshot command	467
mmdf command	470
mmdiag command	473
mmeditacl command	478
mmedquota command	481
mmexportfs command	485
mmfsck command.	487
mmfsctl command.	496
mmgetacl command	500
mmgetstate command	503
mmhadoopctl command.	506
mmimgbbackup command	508
mmimgrestore command	511
mmimportfs command	513
mmlinkfileset command	517
mmlsattr command	519
mmlscallback command.	522
mmlscluster command	524
mmlsconfig command	526
mmlsdisk command	528
mmlsfileset command	532
mmlsfs command	536
mmlslicense command	540
mmlsmgr command	542
mmlsmount command	544
mmlsnodeclass command	546
mmlsnsd command	549
mmlspolicy command	552
mmlspool command	554
mmlsqos command	556
mmlsquota command	559
mmlssnapshot command	563
mmmigratefs command	566
mmmount command	568
mmnfs command	570
mmnsdiscover command	579
mmobj command	581
mmperfmom command	592
mmppmon command	602
mmprotocoltrace command.	607
mmpsnap command	611
mmputacl command	614
mmquotaoff command	617
mmquotaon command	619
mmremotecenter command	621
mmremotefs command	624
mmrepquota command	627
mmrestoreconfig command.	631
mmrestorefs command	635
mmrestripefile command	639
mmrestripefs command	642
mmrpldisk command.	648
mmsdrrestore command.	655
mmsetquota command	657
mmshutdown command.	661
mmsmb command.	663
mmsnapdir command	674

mmstartup command	678
mmtracectl command	680
mmumount command	684
mmunlinkfileset command	687
mmuserauth command	690
mmwinservctl command	709
spectrumscale command	711

Chapter 12. GPFS programming

interfaces 723

gpfs_acl_t structure	726
gpfs_clone_copy() subroutine	727
gpfs_clone_snap() subroutine	729
gpfs_clone_split() subroutine	731
gpfs_clone_unsnap() subroutine	733
gpfs_close_inodescan() subroutine	735
gpfs_cmp_fssnapid() subroutine	736
gpfs_declone() subroutine	738
gpfs_direntx_t structure	740
gpfs_direntx64_t structure	742
gpfs_fcntl() subroutine	744
gpfs_fgetattrs() subroutine	747
gpfs_fputattrs() subroutine	749
gpfs_fputattrswithpathname() subroutine	751
gpfs_free_fssnaphandle() subroutine	753
gpfs_fssnap_handle_t structure	754
gpfs_fssnap_id_t structure	755
gpfs_fstat() subroutine	756
gpfs_get_fsnam_from_fssnaphandle() subroutine	758
gpfs_get_fssnaphandle_by_fssnapid() subroutine	759
gpfs_get_fssnaphandle_by_name() subroutine	761
gpfs_get_fssnaphandle_by_path() subroutine	763
gpfs_get_fssnapid_from_fssnaphandle() subroutine	765
gpfs_get_pathname_from_fssnaphandle() subroutine	767
gpfs_get_snapdirname() subroutine	769
gpfs_get_snapname_from_fssnaphandle() subroutine	771
gpfs_getacl() subroutine	773
gpfs_iattr_t structure	775
gpfs_iattr64_t structure	778
gpfs_iclose() subroutine	782
gpfs_ifile_t structure	784
gpfs_igetattrs() subroutine	785
gpfs_igetattrsx() subroutine	787
gpfs_igetfilesetname() subroutine	789
gpfs_igetstoragepool() subroutine	791
gpfs_iopen() subroutine	793
gpfs_iopen64() subroutine	795
gpfs_iputattrsx() subroutine	797
gpfs_iread() subroutine	800
gpfs_ireaddir() subroutine	802
gpfs_ireaddir64() subroutine	804
gpfs_ireadlink() subroutine	806
gpfs_ireadlink64() subroutine	808
gpfs_ireadx() subroutine	810
gpfs_iscan_t structure	813
gpfs_lib_init() subroutine	814
gpfs_lib_term() subroutine	815
gpfs_next_inode() subroutine	816
gpfs_next_inode64() subroutine	818

gpfs_next_inode_with_xattrs() subroutine	820
gpfs_next_inode_with_xattrs64() subroutine	822
gpfs_next_xattr() subroutine	824
gpfs_opaque_acl_t structure	826
gpfs_open_inodescan() subroutine	827
gpfs_open_inodescan64() subroutine	830
gpfs_open_inodescan_with_xattrs() subroutine	833
gpfs_open_inodescan_with_xattrs64() subroutine	836
gpfs_prealloc() subroutine	839
gpfs_putacl() subroutine	841
gpfs_quotactl() subroutine	843
gpfs_quotaInfo_t structure	846
gpfs_seek_inode() subroutine	848
gpfs_seek_inode64() subroutine	850
gpfs_stat() subroutine	852
gpfs_stat_inode() subroutine	854
gpfs_stat_inode64() subroutine	856
gpfs_stat_inode_with_xattrs() subroutine	858
gpfs_stat_inode_with_xattrs64() subroutine	860
gpfsFcntlHeader_t structure	862
gpfsGetFilesetName_t structure	863
gpfsGetReplication_t structure	864
gpfsGetSetXAttr_t structure	866
gpfsGetSnapshotName_t structure	868
gpfsGetStoragePool_t structure	869
gpfsListXAttr_t structure	870
gpfsRestripeData_t structure	871
gpfsSetReplication_t structure	873
gpfsSetStoragePool_t structure	875

Chapter 13. GPFS user exits 877

mmsdrbackup user exit	878
nsddevices user exit	879
syncfsconfig user exit	880

Chapter 14. Considerations for GPFS

applications 881

Exceptions to Open Group technical standards	881
Determining if a file system is controlled by GPFS	881
GPFS exceptions and limitations to NFS V4 ACLs	882
Linux ACLs and extended attributes	882
General NFS V4 Linux exceptions and limitations	883
Considerations for the use of direct I/O (O_DIRECT)	883

Chapter 15. File system format changes between versions of GPFS . 885

Accessibility features for IBM

Spectrum Scale 889

Accessibility features	889
Keyboard navigation	889
IBM and accessibility	889

Notices 891

Trademarks	893
Terms and conditions for product documentation	893
IBM Online Privacy Statement	894

Glossary	895
Index	901

Tables

1. IBM Spectrum Scale library information units	xi	17. ACL permissions required to work on files and directories, while using SMB protocol (table 2 of 2)	204
2. Conventions	xiv	18. ACL permissions required to work on files and directories, while using NFS protocol (table 1 of 2)	205
3. Configuration attributes on the mmchconfig command	12	19. ACL permissions required to work on files and directories, while using NFS protocol (table 2 of 2)	205
4. COMPRESSED and illCompressed indicators	38	20. Commands and reference to manage ACL tasks	208
5. Set QoS classes to unlimited	41	21. ACL options that are available to manipulate object read ACLs	212
6. Allocate the available IOPS	41	22. GPFS commands	217
7. Authentication requirements for each file access protocol	91	23. Global events and supported parameters	231
8. Object services and object protocol nodes	124	24. Local events and supported parameters	232
9. Object input behavior in <code>unified_mode</code>	137	25. key-value	296
10. Configuration options for [swift-constraints] in <code>swift.conf</code>	151	26. key-value	297
11. Configurable options for [DEFAULT] in <code>object-server-sof.conf</code>	153	27. Restoring a global snapshot	636
12. Configurable options for [capabilities] in <code>spectrum-scale-object.conf</code>	155	28. Restoring a fileset snapshot	636
13. Configuration options for [DEFAULT] in <code>spectrum-scale-objectizer.conf</code>	155	29. GPFS programming interfaces	723
14. Configuration options for [IBMOBJECTIZER-LOGGER] in <code>spectrum-scale-objectizer.conf</code>	155	30. GPFS user exits	877
15. Removal of a file with ACL entries DELETE and DELETE_CHILD	193		
16. ACL permissions required to work on files and directories, while using SMB protocol (table 1 of 2)	203		

About this information

This edition applies to IBM Spectrum Scale™ version 4.2 for AIX®, Linux, and Windows.

IBM Spectrum Scale is a file management infrastructure, based on IBM® General Parallel File System (GPFS™) technology, that provides unmatched performance and reliability with scalable access to critical file data.

To find out which version of IBM Spectrum Scale is running on a particular AIX node, enter:

```
lslpp -l gpfs\*
```

To find out which version of IBM Spectrum Scale is running on a particular Linux node, enter:

```
rpm -qa | grep gpfs
```

To find out which version of IBM Spectrum Scale is running on a particular Windows node, open the **Programs and Features** control panel. The IBM Spectrum Scale installed program name includes the version number.

Which IBM Spectrum Scale information unit provides the information you need?

The IBM Spectrum Scale library consists of the information units listed in Table 1.

To use these information units effectively, you must be familiar with IBM Spectrum Scale and the AIX, Linux, or Windows operating system, or all of them, depending on which operating systems are in use at your installation. Where necessary, these information units provide some background information relating to AIX, Linux, or Windows; however, more commonly they refer to the appropriate operating system documentation.

Note: Throughout this documentation, the term “Linux” refers to all supported distributions of Linux, unless otherwise specified.

Table 1. IBM Spectrum Scale library information units

Information unit	Type of information	Intended users
<i>IBM Spectrum Scale: Administration and Programming Reference</i>	This information unit explains how to do the following: <ul style="list-style-type: none">• Use the commands, programming interfaces, and user exits unique to GPFS• Manage clusters, file systems, disks, and quotas• Export a GPFS file system using the Network File System (NFS) protocol	System administrators or programmers of GPFS systems

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
<i>IBM Spectrum Scale: Advanced Administration Guide</i>	<p>This information unit explains how to use the following advanced features of GPFS:</p> <ul style="list-style-type: none"> • Accessing GPFS file systems from other GPFS clusters • Policy-based data management for GPFS • Creating and maintaining snapshots of GPFS file systems • Establishing disaster recovery for your GPFS cluster • Monitoring GPFS I/O performance with the mmpmon command • Miscellaneous advanced administration topics 	System administrators or programmers seeking to understand and use the advanced features of GPFS
<i>IBM Spectrum Scale: Concepts, Planning, and Installation Guide</i>	<p>This information unit provides information about the following topics:</p> <ul style="list-style-type: none"> • Introducing GPFS • GPFS architecture • Planning concepts for GPFS • Installing GPFS • Migration, coexistence and compatibility • Applying maintenance • Configuration and tuning • Uninstalling GPFS 	System administrators, analysts, installers, planners, and programmers of GPFS clusters who are very experienced with the operating systems on which each GPFS cluster is based

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
<i>IBM Spectrum Scale: Data Management API Guide</i>	<p>This information unit describes the Data Management Application Programming Interface (DMAPI) for GPFS.</p> <p>This implementation is based on The Open Group's System Management: Data Storage Management (XDMS) API Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X specification. The implementation is compliant with the standard. Some optional features are not implemented.</p> <p>The XDMS DMAPI model is intended mainly for a single-node environment. Some of the key concepts, such as sessions, event delivery, and recovery, required enhancements for a multiple-node environment such as GPFS.</p> <p>Use this information if you intend to write application programs to do the following:</p> <ul style="list-style-type: none"> • Monitor events associated with a GPFS file system or with an individual file • Manage and maintain GPFS file system data 	Application programmers who are experienced with GPFS systems and familiar with the terminology and concepts in the XDMS standard
<i>IBM Spectrum Scale: Problem Determination Guide</i>	This information unit contains explanations of GPFS error messages and explains how to handle problems you may encounter with GPFS.	System administrators of GPFS systems who are experienced with the subsystems used to manage disks and who are familiar with the concepts presented in the <i>IBM Spectrum Scale: Concepts, Planning, and Installation Guide</i>

Prerequisite and related information

For updates to this information, see IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html).

For the latest support information, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Conventions used in this information

Table 2 on page xiv describes the typographic conventions used in this information. UNIX file name conventions are used throughout this information.

Note: Users of IBM Spectrum Scale for Windows must be aware that on Windows, UNIX-style file names need to be converted appropriately. For example, the GPFS cluster configuration data is stored in the `/var/mmfs/gen/mmsdrfs` file. On Windows, the UNIX namespace starts under the `%SystemDrive%\cygwin64` directory, so the GPFS cluster configuration data is stored in the `C:\cygwin64\var\mmfs\gen\mmsdrfs` file.

Table 2. Conventions

Convention	Usage
bold	<p>Bold words or characters represent system elements that you must use literally, such as commands, flags, values, and selected menu options.</p> <p>Depending on the context, bold typeface sometimes represents path names, directories, or file names.</p>
<u>bold underlined</u>	<u>bold underlined</u> keywords are defaults. These take effect if you do not specify a different keyword.
constant width	<p>Examples and information that the system displays appear in constant-width typeface.</p> <p>Depending on the context, constant-width typeface sometimes represents path names, directories, or file names.</p>
<i>italic</i>	<p><i>Italic</i> words or characters represent variable values that you must supply.</p> <p><i>Italics</i> are also used for information unit titles, for the first use of a glossary term, and for general emphasis in text.</p>
<key>	Angle brackets (less-than and greater-than) enclose the name of a key on the keyboard. For example, <Enter> refers to the key on your terminal or workstation that is labeled with the word <i>Enter</i> .
\	<p>In command examples, a backslash indicates that the command or coding example continues on the next line. For example:</p> <pre>mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \ -E "PercentTotUsed < 85" -m p "FileSystem space used"</pre>
{item}	Braces enclose a list from which you must choose an item in format and syntax descriptions.
[item]	Brackets enclose optional items in format and syntax descriptions.
<Ctrl-x>	The notation <Ctrl-x> indicates a control character sequence. For example, <Ctrl-c> means that you hold down the control key while pressing <c>.
item...	Ellipses indicate that you can repeat the preceding item one or more times.
	<p>In <i>synopsis</i> statements, vertical lines separate a list of choices. In other words, a vertical line means <i>Or</i>.</p> <p>In the left margin of the document, vertical lines indicate technical changes to the information.</p>

How to send your comments

Your feedback is important in helping us to produce accurate, high-quality information. If you have any comments about this information or any other IBM Spectrum Scale documentation, send your comments to the following e-mail address:

mhvrcfs@us.ibm.com

Include the publication title and order number, and, if applicable, the specific location of the information about which you have comments (for example, a page number or a table number).

To contact the IBM Spectrum Scale development organization, send your comments to the following e-mail address:

gpfs@us.ibm.com

Summary of changes

This topic summarizes changes to the IBM Spectrum Scale licensed program and the IBM Spectrum Scale library. Within each information unit in the library, a vertical line (|) to the left of text and illustrations indicates technical changes or additions made to the previous edition of the information.

Summary of changes for IBM Spectrum Scale version 4 release 2 as updated, November 2015

Changes to this release of the IBM Spectrum Scale licensed program and the IBM Spectrum Scale library include the following:

Cluster Configuration Repository (CCR): Backup and restore

You can backup and restore a cluster that has Cluster Configuration Repository (CCR) enabled. In the **mmsdrbackup** user exit, the type of backup that is created depends on the configuration of the cluster. If the Cluster Configuration Repository (CCR) is enabled, then a CCR backup is created. Otherwise, a **mmsdrfs** backup is created. In the **mmsdrrestore** command, if the configuration file is a Cluster Configuration Repository (CCR) backup file, then you must specify the **-a** option. All the nodes in the cluster are restored.

Changes in IBM Spectrum Scale for object storage

Object capabilities

Object capabilities describe the object protocol features that are configured in the IBM Spectrum Scale cluster such as unified file and object access, multi-region object deployment, and S3 API emulation. For more information, see the following topics:

- *Object capabilities in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Managing object capabilities in IBM Spectrum Scale: Administration and Programming Reference*

Storage policies for object storage

Storage policies enable segmenting of the object storage within a single cluster for various use cases. Currently, OpenStack Swift supports storage policies that allow you to define the replication settings and location of objects in a cluster. IBM Spectrum Scale enhances storage policies to add compression and unified file and object access functions for object storage. For more information, see the following topics:

- *Storage policies for object storage in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Mapping of storage policies to filesets in IBM Spectrum Scale: Administration and Programming Reference*
- *Administering storage policies for object storage in IBM Spectrum Scale: Administration and Programming Reference*

Multi-region object deployment

The main purpose of the object protocol is to enable the upload and download of object data. When clients have a fast connection to the cluster, the network delay is minimal. However, when client access to object data is over a WAN or a high-latency network, the network can introduce an unacceptable delay and affect quality-of-service metrics. To improve that response time, you can create a replica of the data in a cluster closer to the clients using the active-active multi-region replication support in OpenStack Swift. Multi-region can also be used to distribute the object load over several clusters to reduce contention in the file system. For more information, see the following topics:

- *Overview of multi-region object deployment in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Planning for multi-region object deployment in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Enabling multi-region object deployment initially in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Adding a region in a multi-region object deployment in IBM Spectrum Scale: Administration and Programming Reference*
- *Administering a multi-region object deployment environment in IBM Spectrum Scale: Administration and Programming Reference*

Unified file and object access

Unified file and object access allows users to access the same data as an object and as a file. Data can be stored and retrieved through IBM Spectrum Scale for object storage or as files from POSIX, NFS, and SMB interfaces. For more information, see the following topics:

- *Unified file and object access overview in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Planning for unified file and object access in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Installing and using unified file and object access in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Unified file and object access in IBM Spectrum Scale in IBM Spectrum Scale: Administration and Programming Reference*

S3 access control lists (ACLs) support

IBM Spectrum Scale for object storage supports S3 access control lists (ACLs) on buckets and objects. For more information, see *Managing OpenStack access control lists using S3 API emulation in IBM Spectrum Scale: Administration and Programming Reference*.

Changes in IBM Spectrum Scale for Linux on z Systems™

- Compression support
- AFM-based Async Disaster Recovery (AFM DR) support
- IBM Spectrum Protect™ Backup-Archive and Space Management client support
- Support for all editions:
 - Express®
 - Standard
 - Advanced (without encryption)

For more information about current requirements and limitations of IBM Spectrum Scale for Linux on z Systems, see Q2.25 of IBM Spectrum Scale FAQ.

Change in AFM-based Async Disaster Recovery (AFM DR)

- Support for IBM Spectrum Scale for Linux on z Systems

File compression

With file compression, you can reclaim some of the storage space occupied by infrequently accessed files. Run the **mmchattr** command or the **mmapplypolicy** command to identify and compress a few files or many files. Run file compression synchronously or defer it. If you defer it, you can run the **mmrestripefile** or **mmrestripefs** to complete the compression. You can decompress files with the same commands used to compress files. When a compressed file is read it is decompressed on the fly and remains compressed on disk. When a compressed file is overwritten, the parts of the file that overlap with the changed data are decompressed on disk synchronously in the granularity of ten data blocks. File compression in this release is designed to

be used only for compressing cold data or write-once objects and files. Compressing other types of data can result in performance degradation. File compression uses the zlib data compression library and favors saving space over speed.

GUI servers

The IBM Spectrum Scale system provides a GUI that can be used for managing and monitoring the system. Any server that provides this GUI service is referred to as a GUI server. If you need GUI service in the system, designate at least two nodes as GUI servers in the cluster. A maximum of three nodes can be designated as GUI servers. For more information on installing IBM Spectrum Scale using the GUI, see *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

IBM Spectrum Scale management GUI

The management GUI helps to manage and monitor the IBM Spectrum Scale system. You can perform the following tasks through management GUI:

- Monitoring the performance of the system based on various aspects
- Monitoring system health
- Managing file systems
- Creating filesets and snapshots
- Managing Objects and NFS and SMB data exports
- Creating administrative users and defining roles for the users
- Creating object users and defining roles for them
- Defining default, user, group, and fileset quotas
- Monitoring the capacity details at various levels such as file system, pools, filesets, users, and user groups

Hadoop Support for IBM Spectrum Scale

IBM Spectrum Scale has been extended to work seamlessly in the Hadoop ecosystem and is available through a feature called File Placement Optimizer (FPO). Storing your Hadoop data using FPO allows you to gain advanced functions and the high I/O performance required for many big data operations. FPO provides Hadoop compatibility extensions to replace HDFS in a Hadoop ecosystem, with no changes required to Hadoop applications.

You can deploy a IBM Spectrum Scale using FPO as a file system platform for big data analytics. The topics in this guide covers a variety of Hadoop deployment architectures, including IBM BigInsights®, Platform Symphony®, or with a Hadoop distribution from another vendor to work with IBM Spectrum Scale.

IBM Spectrum Scale offers two kinds of interfaces for Hadoop applications to access File System data. One is IBM Spectrum Scale connector, which aligns with Hadoop Compatible File System architecture and APIs. The other is HDFS protocol, which provides a HDFS compatible interfaces.

For more information, see the following sections in the *IBM Spectrum Scale: Advanced Administration Guide*:

- *Hadoop support for IBM Spectrum Scale*
- *Configuring FPO*
- *Hadoop connector*
- *HDFS protocol*

IBM Spectrum Scale installation GUI

You can use the installation GUI to install the IBM Spectrum Scale system. For more information on how to launch the GUI installer, see the *Installing IBM Spectrum Scale using the graphical user interface (GUI)* section in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Performance Monitoring Tool using the Installation Kit

The usage statement and optional arguments have changed during the installation of the toolkit. The new usage statement with options is as follows:

```
spectrumscale config perfmon [-h] [-l] [-r {on,off}]
```

For more information, see *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Protocols cluster disaster recovery (DR)

You can use the **mmcesdr** command to perform DR setup, failover, failback, backup, and restore actions. Protocols cluster DR uses the capabilities of Active File Management based Async Disaster Recovery (AFM DR) to provide a solution that allows an IBM Spectrum Scale cluster to fail over to another cluster and fail back, and backup and restore the protocol configuration information in cases where a secondary cluster is not available. For more information, see *Protocols cluster disaster recovery* in *IBM Spectrum Scale: Advanced Administration Guide*.

Quality of Service for I/O operations (QoS)

You can use the QoS capability to prevent I/O-intensive, long-running GPFS commands, called *maintenance commands*, from dominating file system performance and significantly delaying normal tasks that also compete for I/O resources. Determine the maximum capacity of your file system in I/O operations per second (IOPS) with the new **mmisqos** command. With the new **mmchqos** command, assign a smaller share of IOPS to the QoS **maintenance** class, which includes all the maintenance commands. Maintenance command instances that are running at the same time compete for the IOPS allocated to the **maintenance** class, and are not allowed to exceed that limit.

Security mode for new clusters

Starting with IBM Spectrum Scale V4.2, the default security mode for new clusters is AUTHONLY. The **mmcrcluster** command sets the security mode to AUTHONLY when it creates the cluster and automatically generates a public/private key pair for authenticating the cluster. In the AUTHONLY security mode, the sending and receiving nodes authenticate each other with a TLS handshake and then close the TLS connection. Communication continues in the clear. The nodes do not encrypt transmitted data and do not check data integrity.

In IBM Spectrum Scale V4.1 or earlier, the default security mode is EMPTY. If you update a cluster from IBM Spectrum Scale V4.1 to V4.2 or later by running **mmchconfig release=LATEST**, the command checks the security mode. If the mode is EMPTY, the command issues a warning message but does not change the security mode of the cluster.

Snapshots

You can display information about a global snapshot without displaying information about fileset snapshots with the same name. You can display information about a fileset snapshot without displaying information about other snapshots that have the same name but are snapshots of other filesets.

spectrumscale Options

The **spectrumscale** command options for installing GPFS and deploying protocols have changed to remove **config enable** and to add **config perf**. For more information, see *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

New options have been added to **spectrumscale setup** and **spectrumscale deploy** to disable prompting for the encryption/decryption secret. Note that if **spectrumscale setup --storesecret** is used, passwords will not be secure. New properties have been added to **spectrumscale cofig object** for setting password data instead of doing so through **enable object**. For more information, see *IBM Spectrum Scale: Administration and Programming Reference*.

The **spectrumscale** options for managing share ACLs have been added. For more information, see *IBM Spectrum Scale: Administration and Programming Reference*.

ssh and scp wrapper scripts

Starting with IBM Spectrum Scale V4.2, a cluster can be configured to use **ssh** and **scp** wrappers. The wrappers allow GPFS to run on clusters where remote root login through **ssh** is disabled. For more information, see the help topic "Running IBM Spectrum Scale without remote root login" in the *IBM Spectrum Scale: Administration and Programming Reference*.

Documented commands, structures, and subroutines

The following lists the modifications to the documented commands, structures, and subroutines:

New commands

The following commands are new:

- **mmcallhome**
- **mmcesdr**
- **mmchqos**
- **mmlsqos**

New structures

There are no new structures.

New subroutines

There are no new subroutines.

Changed commands

The following commands were changed:

- **mmadddisk**
- **mmaddnode**
- **mmapplypolicy**
- **mmauth**
- **mmbackup**
- **mmces**
- **mmchattr**
- **mmchcluster**
- **mmchconfig**
- **mmchdisk**
- **mmcheckquota**
- **mmchnode**
- **mmcrcluster**
- **mmdefragfs**
- **mmdeldisk**
- **mmdelfileset**
- **mmdelsnapshot**
- **mmdf**
- **mmfileid**
- **mmfsck**
- **mmlsattr**
- **mmlscluster**
- **mmlsconfig**
- **mmlssnapshot**
- **mmnfs**
- **mmobj**
- **mmperfmon**
- **mmprotocoltrace**
- **mmremotefs**
- **mmrestripefile**
- **mmrestripefs**
- **mmrpldisk**
- **mmsdrbackup**

- **mmsdrrestore**
- **mmsmb**
- **mmuserauth**
- **spectrumscale**

Changed structures

There are no changed structures.

Changed subroutines

There are no changed subroutines.

Deleted commands

There are no deleted commands.

Deleted structures

There are no deleted structures.

Deleted subroutines

There are no deleted subroutines.

Messages

The following lists the new, changed, and deleted messages:

New messages

6027-2354, 6027-2355, 6027-2356, 6027-2357, 6027-2358, 6027-2359, 6027-2360, 6027-2361,
6027-2362, 6027-3913, 6027-3914, 6027-3107, 6027-4016, 6027-3317, 6027-3318, 6027-3319,
6027-3320, 6027-3405, 6027-3406, 6027-3582, 6027-3583, 6027-3584, 6027-3585, 6027-3586,
6027-3587, 6027-3588, 6027-3589, 6027-3590, 6027-3591, 6027-3592, 6027-3593

Changed messages

6027-2299, 6027-887, 6027-888

Deleted messages

None.

Chapter 1. Performing GPFS administration tasks

Before you perform GPFS administration tasks, review topics such as getting started with GPFS, requirements for administering a GPFS file system, and common command principles.

For information on getting started with GPFS, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*. This includes:

1. Installing GPFS
2. GPFS cluster creation considerations
3. Configuring and tuning your system for GPFS
4. Starting GPFS
5. Network Shared Disk creation considerations
6. File system creation considerations

This guide covers the administration and maintenance of GPFS and your file systems, and includes the following topics:

1. “Requirements for administering a GPFS file system” and “Common GPFS command principles” on page 3
2. Chapter 2, “Managing the GPFS cluster,” on page 7
3. Chapter 3, “Managing file systems,” on page 27
4. Chapter 7, “Managing disks,” on page 157
5. Chapter 8, “Managing GPFS quotas,” on page 169
6. Chapter 10, “Managing GPFS access control lists,” on page 187
7. Chapter 11, “GPFS commands,” on page 217
8. Chapter 12, “GPFS programming interfaces,” on page 723
9. Chapter 13, “GPFS user exits,” on page 877
10. Chapter 14, “Considerations for GPFS applications,” on page 881
11. Chapter 15, “File system format changes between versions of GPFS,” on page 885

For more advanced GPFS administration topics, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Requirements for administering a GPFS file system

Root authority is required to perform all GPFS administration tasks except those with a function limited to listing certain GPFS operating characteristics or modifying individual user file attributes.

On Windows, root authority normally means users in the Administrators group. However, for clusters with both Windows and UNIX nodes, only the special Active Directory domain user **root** qualifies as having root authority for the purposes of administering GPFS. For more information on GPFS prerequisites, see the topic *Installing GPFS prerequisites* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

The GPFS commands are designed to maintain the appropriate environment across all nodes in the cluster. To achieve this goal, the GPFS commands use the remote shell and remote file copy commands that you specify on either the **mmcrcluster** or the **mmchcluster** command.

The default remote commands are **ssh** and **scp**, but you can designate any other remote commands provided they have compatible syntax.

In principle, you can issue GPFS administration commands from any node in the cluster. The nodes that you plan to use for administering GPFS must be able to execute remote shell commands on themselves and on any other node in the cluster. They must do so without the use of a password and without producing any extraneous messages. Similarly, the nodes on which the GPFS commands are issued must be able to copy files to and from any other node in the cluster. And the nodes must do so without the use of a password and without producing any extraneous messages.

The way the passwordless access is achieved depends on the particular remote execution program and authentication mechanism that is used. For example, for **rsh** and **rsh**, you might need a properly configured **.rhosts** file in the root user's home directory on each node in the GPFS cluster. If the remote program is **ssh**, you can use private identity files that do not have a password. Or, if the identity file is password-protected, you can use the **ssh-agent** utility to establish an authorized session before you issue **mm** commands.

You can avoid configuring your GPFS nodes to allow remote access to the root user ID, by using sudo wrapper scripts to run GPFS administrative commands. See "Running IBM Spectrum Scale without remote root login" on page 21.

GPFS does not need to know which nodes are being used for administration purposes. It is the administrator's responsibility to issue **mm** commands only from nodes that are properly configured and can access the rest of the nodes in the cluster.

Note: If your cluster includes Windows nodes, you must designate **ssh** and **scp** as the remote communication program.

adminMode configuration attribute

GPFS recognizes the **adminMode** configuration attribute. It specifies whether all nodes in the cluster will be used for issuing GPFS administration commands or just a subset of the nodes.

The **adminMode** attribute is set with the **mmchconfig** command and can have one of two values:

a11ToA11

Indicates that all nodes in the cluster can be used for running GPFS administration commands and that all nodes are able to execute remote commands on any other node in the cluster without the need of a password.

The major advantage of this mode of operation is that GPFS can automatically recover missing or corrupted configuration files in almost all circumstances. The major disadvantage is that all nodes in the cluster must have root level access to all other nodes.

central

Indicates that only a subset of the nodes will be used for running GPFS commands and that only those nodes will be able to execute remote commands on the rest of the nodes in the cluster without the need of a password.

The major advantage of this mode of administration is that the number of nodes that must have root level access to the rest of the nodes is limited and can be as low as one. The disadvantage is that GPFS may not be able to automatically recover from loss of certain configuration files. For example, if the SSL key files are not present on some of the nodes, the operator may have to intervene to recover the missing data. Similarly, it may be necessary to shut down GPFS when adding new quorum nodes. If an operator intervention is needed, you will see appropriate messages in the GPFS log or on the screen.

Note List:

1. If the GPFS cluster is configured to support Clustered NFS (CNFS), all CNFS member nodes must belong to the subset of nodes that are able to execute remote commands without the need of a password.

2. If Tivoli® Storage Manager (TSM) server is used to back up the GPFS file system data, the nodes that are used as TSM clients must belong to the subset of nodes that are able to execute remote commands without the need of a password.
3. Windows GPFS clusters typically use **central** mode. **allToAll** mode requires that the GPFS Administrative service (mmwinserv) be configured to run as the special domain root account. For more information, see the *Installing GPFS on Windows nodes* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Clusters created with the GPFS 3.3 or later level of the code have **adminMode** set to **central** by default. Clusters migrated from GPFS 3.2 or earlier versions will continue to operate as before and will have **adminMode** set to **allToAll**.

You can change the mode of operations at any time with the help of the **mmchconfig** command. For example, to switch the mode of administration from **allToAll** to **central**, issue:

```
mmchconfig adminMode=central
```

Use the **mmlsconfig adminMode** command to display the mode of administration currently in effect for the cluster.

Common GPFS command principles

There are some common principles that you should keep in mind when you are running GPFS commands.

Those principles include:

- Unless otherwise noted, GPFS commands can be run from any node in the cluster. Exceptions are commands that are not supported in a particular operating system environment. Certain commands may additionally require the affected file system to be mounted.
- GPFS supports the "no" prefix on all Boolean type long (or dash-dash) options.

Specifying nodes as input to GPFS commands

Many GPFS commands accept a node or multiple nodes as part of their input, using the **-N** flag.

Nodes can be specified to GPFS commands in a variety of ways:

Node

A representation of an individual node, which can be any of these:

- Short GPFS administration node interface name.
- Long GPFS administration node interface name.
- Short GPFS daemon node interface name.
- Long GPFS daemon node interface name.
- IP address corresponding to the GPFS daemon node interface.
- GPFS node number.

Node - Node

A node range, indicated by specifying two node numbers separated by a hyphen (-), with the first node number being less than or equal to the second node number. For example, node range **3-8** specifies the nodes with node numbers 3, 4, 5, 6, 7, and 8.

NodeClass

A set of nodes that are grouped into system-defined node classes or user-defined node classes. The system-defined node classes that are known to GPFS are:

all

All of the nodes in the GPFS cluster.

clientnodes

All nodes that do not participate in file system administration activities.

localhost

The node on which the command is running.

managernodes

All nodes in the pool of nodes from which file system managers and token managers are selected.

mount

For commands involving a file system, all of the local nodes on which the file system is mounted (nodes in remote clusters are always excluded, even when they mount the file system in question).

nonquorumnodes

All of the non-quorum nodes in the GPFS cluster.

nsdnodes

All of the NSD server nodes in the GPFS cluster.

quorumnodes

All of the quorum nodes in the GPFS cluster.

User-defined node classes are created with the **mmcrnodeclass** command. After a node class is created, it can be specified as an argument on commands that accept the **-N NodeClass** option. User-defined node classes are managed with the **mmchnodeclass**, **mmdelnodeclass**, and **mmlsnodeclass** commands.

NodeFile

A file that contains a list of nodes. A node file can contain individual nodes or node ranges.

For commands operating on a file system, the stripe group manager node is always implicitly included in the node list. Not every GPFS command supports all of the node specification options described in this topic. To learn what kinds of node specifications are supported by a particular GPFS command, see the relevant command description in Chapter 11, "GPFS commands," on page 217.

Stanza files

The input to a number of GPFS commands can be provided in a file organized in a stanza format.

A stanza is a series of whitespace-separated tokens that can span multiple lines. The beginning of a stanza is indicated by the presence of a stanza identifier as the first token on a line. Stanza identifiers consist of the % (percent sign) character, followed by a keyword, and ending with the : (colon) character. For example, **%nsd:** indicates the beginning of an NSD stanza.

A stanza identifier is followed by one or more stanza clauses describing different properties of the object. A stanza clause is defined as an *Attribute=value* pair.

Lines that start with the # (pound sign) character are considered comment lines and are ignored. Similarly, you can imbed inline comments following a stanza clause; all text after the # character is considered a comment.

The end of a stanza is indicated by one of the following:

- a line that represents the beginning of a new stanza
- a blank line
- a non-comment line that does not contain the = character

GPFS recognizes a number of stanzas:

- %nsd:**
NSD stanza
- %pdisk:**
Physical disk stanza
- %vdisk:**
Virtual disk stanza
- %da:**
Declustered array stanza
- %rg:**
Recovery group stanza

The details are documented under the corresponding commands.

For more information about the GPFS Native RAID commands that use stanzas, see *IBM Spectrum Scale RAID: Administration*.

A stanza file can contain multiple types of stanzas. Commands that accept input in the form of stanza files expect the stanzas to be syntactically correct but will ignore stanzas that are not applicable to the particular command. Similarly, if a particular stanza clause has no meaning for a given command, it is ignored.

For backward compatibility, a stanza file may also contain traditional NSD descriptors, although their use is discouraged.

Here is what a stanza file may look like:

```
# Sample file containing two NSD stanzas

# Example for an NSD stanza with imbedded comments
%nsd: nsd=DATA5      # my name for this NSD
      device=/dev/hdisk5 # device name on node k145n05
      usage=dataOnly
      # List of server nodes for this disk
      servers=k145n05,k145n06
      failureGroup=2
      pool=dataPoolA

# Example for a directly attached disk; most values are allowed to default
%nsd: nsd=DATA6  device=/dev/hdisk6  failureGroup=3
```

Listing active GPFS commands

- | You can list the active GPFS commands that are running on the file system manager node.
- | Most GPFS commands run within the GPFS daemon on the file system manager node. Even if you start a command on another node of the cluster, the node typically sends the command to the file system manager node to be executed. (Two exceptions are the **mmdiag** command and the **mmfsadm dump** command, which run on the node where they were started.)
- | To list the active commands on the file system manager node, follow these steps:
 - | 1. Enter the **mmlsmgr** command with no parameters to discover which node is the file system manager node. In the following example, the **mmlsmgr** command reports that node05 is the file system manager node:

```
| # mmlsmgr
| file system      manager node
| -----
| gpfs1            192.168.145.14 (node05)
|
| Cluster manager node: 192.168.145.13 (node03)
```

| 2. Go to the command console on the file system manager node and enter **mmdiag --commands**:

```
| # mmdiag --commands
| === mmdiag: commands ===
| CrHashTable 0x1167A28F0 n 2
|   cmd sock 24 cookie 2233688162 owner 38076509 id 0x3FE6046C2700000D(#13) uses 1
|   type 76 start 1460415325.957724 flags 0x106 SG none line 'mmdiag --commands'
|   cmd sock 12 cookie 521581069 owner 57606185 id 0x3FE6046C2700000C(#12) uses 1
|   type 13 start 1460415323.336314 flags 0x117 SG gpfs1 line 'mmrestripefs /dev/business1 -m'
```

| The output indicates that two GPFS commands are running: the **mmdiag --commands** command that you just entered and the **mmrestripefs** command, which was started from another node.

| **Note:** The output contains two lines about active commands. Each line begins with the term **cmd** and wraps to the next line. You might be interested in the following fields:

| **start** The system time at which the command was received.

| **SG** The name of the file system, or None.

| **line** The command as received by the GPFS daemon.

| The remaining input is detailed debugging data that is used for product support.

Chapter 2. Managing the GPFS cluster

There are several tasks involved in managing your GPFS cluster. This topic points you to the information you need to get started.

GPFS cluster management tasks include the following.

- “Creating your GPFS cluster”
- “Displaying GPFS cluster configuration information”
- “Specifying nodes as input to GPFS commands” on page 3
- “Adding nodes to a GPFS cluster” on page 8
- “Deleting nodes from a GPFS cluster” on page 9
- “Changing the GPFS cluster configuration data” on page 10
- “Node quorum considerations” on page 23
- “Node quorum with tiebreaker considerations” on page 23
- “Displaying and changing the file system manager node” on page 24
- “Determining how long **mmrestripefs** takes to complete” on page 24
- “Starting and stopping GPFS” on page 25

Note: In IBM Spectrum Scale V4.1.1 and later, many of these tasks can also be handled by the **spectrumscale** installation toolkit configuration options. For more information on the installation toolkit, see *Using the spectrumscale installation toolkit to perform installation tasks: Explanations and examples* section in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For information on RAID administration, see *IBM Spectrum Scale RAID: Administration*.

Creating your GPFS cluster

You must first create a GPFS cluster by issuing the **mmcrcluster** command.

For more information, see **mmcrcluster command** in *IBM Spectrum Scale: Administration and Programming Reference*.

For details on how GPFS clusters are created and used, see *GPFS cluster creation considerations* topic in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Displaying GPFS cluster configuration information

When managing your GPFS cluster, you can display the current configuration information for the cluster by issuing the **mmlscluster** command.

The command displays:

- The cluster name
- The cluster ID
- GPFS UID domain
- The remote shell command being used
- The remote file copy command being used
- The repository type (CCR or server-based)
- The primary GPFS cluster configuration server (if server-based repository)

- The secondary GPFS cluster configuration server (if server-based repository)
- A list of nodes belonging the GPFS cluster

For each node, the command displays:

- The node number assigned to the node by GPFS
- Daemon node name
- Network IP address
- Admin node name
- Designation, such as whether the node is a quorum node, a manager node, or an snmp_collector node or all of these

To display this information, enter:

```
mmlscluster
```

The system displays information similar to:

```
GPFS cluster information
```

```
=====
```

```
GPFS cluster name:      cluster1.kgn.ibm.com
GPFS cluster id:       680681562214606028
GPFS UID domain:      cluster1.kgn.ibm.com
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:      CCR
```

Node	Daemon node name	IP address	Admin node name	Designation
1	k164n04.kgn.ibm.com	198.117.68.68	k164n04.kgn.ibm.com	quorum
2	k164n05.kgn.ibm.com	198.117.68.69	k164n05.kgn.ibm.com	quorum
3	k164n06.kgn.ibm.com	198.117.68.70	k164sn06.kgn.ibm.com	quorum-manager

For complete usage information, see the “mmlscluster command” on page 524.

Adding nodes to a GPFS cluster

You can add nodes to an existing GPFS cluster by issuing the **mmaddnode** command. The new nodes are available immediately after the successful completion of this command.

You must follow these rules when adding nodes to a GPFS cluster:

- You may issue the command only from a node that already belongs to the GPFS cluster.
- A node may belong to only one GPFS cluster at a time.
- The nodes must be available for the command to be successful. If any of the nodes listed are not available when the command is issued, a message listing those nodes is displayed. You must correct the problem on each node and reissue the command to add those nodes.
- After the nodes are added to the cluster, you must use the **mmchlicense** command to designate appropriate GPFS licenses to the new nodes.

To add node **k164n06** to the GPFS cluster, enter:

```
mmaddnode -N k164n06
```

The system displays information similar to:

```
Mon Aug 9 21:53:30 EDT 2004: 6027-1664 mmaddnode: Processing node k164n06.kgn.ibm.com
mmaddnode: Command successfully completed
mmaddnode: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

To confirm the addition of the nodes, enter:

```
mmlscluster
```

The system displays information similar to:

```
GPFS cluster information
=====
GPFS cluster name: cluster1.kgn.ibm.com
GPFS cluster id: 680681562214606028
GPFS UID domain: cluster1.kgn.ibm.com
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type: server-based
```

```
GPFS cluster configuration servers:
-----
Primary server: k164sn06.kgn.ibm.com
Secondary server: k164n05.kgn.ibm.com
```

```
Node Daemon node name IP address Admin node name Designation
-----
1 k164n04.kgn.ibm.com 198.117.68.68 k164n04.kgn.ibm.com quorum
2 k164n05.kgn.ibm.com 198.117.68.69 k164n05.kgn.ibm.com quorum
3 k164n06.kgn.ibm.com 198.117.68.70 k164sn06.kgn.ibm.com quorum-manager
```

For complete usage information, see “mmaddnode command” on page 245 and “mmlscluster command” on page 524.

Deleting nodes from a GPFS cluster

You can delete nodes from a GPFS cluster by issuing the **mmdelnode** command.

You must follow these rules when deleting nodes:

- A node being deleted cannot be the primary or secondary GPFS cluster configuration server unless you intend to delete the entire cluster. Verify this by issuing the **mmlscluster** command. If a node to be deleted is one of the servers and you intend to keep the cluster, issue the **mmchcluster** command to assign another node as a configuration server before deleting the node.
- A node that is being deleted cannot be designated as an NSD server for any disk in the GPFS cluster, unless you intend to delete the entire cluster. Verify this by issuing the **mmlsnsd** command. If a node that is to be deleted is an NSD server for one or more disks, move the disks to nodes that will remain in the cluster. Issue the **mmchnsd** command to assign new NSD servers for those disks.
- GPFS must be shut down on the nodes being deleted. Issue the **mmsshutdown** command.

To delete the nodes listed in a file called **nodes_to_delete**, issue:

```
mmdelnode -N /tmp/nodes_to_delete
```

where **nodes_to_delete** contains the nodes **k164n01** and **k164n02**. The system displays information similar to:

```
Verifying GPFS is stopped on all affected nodes ...
mmdelnode: Command successfully completed
mmdelnode: 6027-1371 Propagating the cluster configuration data to all
    affected nodes. This is an asynchronous process.
```

To confirm the deletion of the nodes, issue:

```
mmlscluster
```

The system displays information similar to:

```
GPFS cluster information
=====
GPFS cluster name:          cluster1.kgn.ibm.com
```

```
GPFS cluster id:          680681562214606028
GPFS UID domain:         cluster1.kgn.ibm.com
Remote shell command:    /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:         server-based
```

GPFS cluster configuration servers:

```
-----
Primary server:  k164sn06.kgn.ibm.com
Secondary server: k164n05.kgn.ibm.com
```

Node	Daemon node name	IP address	Admin node name	Designation
1	k164n04.kgn.ibm.com	198.117.68.68	k164n04.kgn.ibm.com	quorum
2	k164n05.kgn.ibm.com	198.117.68.69	k164n05.kgn.ibm.com	quorum
3	k164n06.kgn.ibm.com	198.117.68.70	k164sn06.kgn.ibm.com	quorum-manager

For complete usage information, see “`mmdelnode` command” on page 460 and “`mmlscluster` command” on page 524.

Exercise caution when shutting down GPFS on quorum nodes or deleting quorum nodes from the GPFS cluster. If the number of remaining quorum nodes falls below the requirement for a quorum, you will be unable to perform file system operations. For more information on quorum, see the section on *Quorum*, in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Changing the GPFS cluster configuration data

You can use the `mmchcluster` or `mmchconfig` commands to change the configuration attributes.

After you have configured the GPFS cluster, you can change configuration attributes with the `mmchcluster` command or the `mmchconfig` command. For more information, see the following topics:

- The topic *mmchcluster command* in the *IBM Spectrum Scale: Administration and Programming Reference* guide
- The topic *mmchconfig command* in the *IBM Spectrum Scale: Administration and Programming Reference* guide

Use the `mmchcluster` command to do the following tasks:

- Change the name of the cluster.
- Change the remote shell and remote file copy programs to be used by the nodes in the cluster. These commands must adhere to the syntax forms of the `ssh` and `scp` commands, but may implement an alternate authentication mechanism.
- Enable or disable the cluster configuration repository (CCR). For more information, see the *Cluster configuration data files* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

If you are using the traditional server-based (non-CCR) configuration repository, you can also do the following tasks:

- Change the primary or secondary GPFS cluster configuration server nodes. The primary or secondary server may be changed to another node in the GPFS cluster. That node must be available for the command to be successful.

Attention: If during the change to a new primary or secondary GPFS cluster configuration server, one or both of the old server nodes are down, it is imperative that you run the `mmchcluster -p LATEST` command as soon as the old servers are brought back online. Failure to do so may lead to disruption in GPFS operations.

- Synchronize the primary GPFS cluster configuration server node. If an invocation of the `mmchcluster` command fails, you will be prompted to reissue the command and specify `LATEST` on the `-p` option to

synchronize all of the nodes in the GPFS cluster. Synchronization instructs all nodes in the GPFS cluster to use the most recently specified primary GPFS cluster configuration server.

For example, to change the primary server for the GPFS cluster data, enter:

```
mmchcluster -p k164n06
```

The system displays information similar to:

```
mmchcluster -p k164n06
mmchcluster: Command successfully completed
```

To confirm the change, enter:

```
mmlscluster
```

The system displays information similar to:

```
GPFS cluster information
=====
GPFS cluster name:      cluster1.kgn.ibm.com
GPFS cluster id:       680681562214606028
GPFS UID domain:      cluster1.kgn.ibm.com
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:      server-based
```

GPFS cluster configuration servers:

```
-----
Primary server:   k164sn06.kgn.ibm.com
Secondary server: k164n05.kgn.ibm.com
```

Node	Daemon node name	IP address	Admin node name	Designation
1	k164n04.kgn.ibm.com	198.117.68.68	k164n04.kgn.ibm.com	quorum
2	k164n05.kgn.ibm.com	198.117.68.69	k164n05.kgn.ibm.com	quorum
3	k164n06.kgn.ibm.com	198.117.68.70	k164sn06.kgn.ibm.com	quorum-manager

Attention: The **mmchcluster** command, when issued with either the **-p** or **-s** option, is designed to operate in an environment where the current primary and secondary GPFS cluster configuration servers are *not* available. As a result, the command can run without obtaining its regular serialization locks. To assure smooth transition to a new cluster configuration server, no other GPFS commands (**mm...** commands) should be running when the command is issued nor should any other command be issued until the **mmchcluster** command has successfully completed.

For complete usage information, see “mmchcluster command” on page 327 and “mmlscluster command” on page 524.

You may be able to tune your cluster for better performance by re-configuring one or more attributes. Before you change any attributes, consider how the changes will affect the operation of GPFS. For a detailed discussion see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the **mmcrcluster** command.

Table 3 on page 12 details the GPFS cluster configuration attributes which can be changed by issuing the **mmchconfig** command. Variations under which these changes take effect are noted:

1. Take effect immediately and are permanent (**-i**).
2. Take effect immediately but do not persist when GPFS is restarted (**-I**).
3. Require that the GPFS daemon be stopped on all nodes for the change to take effect.
4. May be applied to only a subset of the nodes in the cluster.

Table 3. Configuration attributes on the **mmchconfig** command

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
adminMode Controls password-less access	yes	no	no	no	immediately
atimeDeferredSeconds Update behavior of atime when relatime is enabled	yes	yes	no	yes	if not immediately, on restart of the daemon
autoload Starts GPFS automatically	no	no	no	yes	on reboot of each node
automountDir Name of the automount directory	no	no	yes	no	on restart of the daemon
cesSharedRoot A directory to be used by the CES subsystem.	yes	no	yes (on all CES nodes)	no	immediately
cipherList The security mode of the cluster. This value indicates the level of security that the cluster uses for communications between nodes in the cluster and also for communications between clusters.	no	no	only when changing from AUTHONLY or a cipher to EMPTY mode	no	for new connections
cnfsGrace The number of seconds a CNFS node will deny new client requests after a node failover or failback	yes	no	yes	no	immediately
cnfsMountdPort The port number to be used for rpc.mountd	yes	no	no	no	immediately
cnfsNFSDprocs The number of nfsd kernel threads	yes	no	no	no	if not immediately, on restart of the daemon
cnfsReboot Determines whether the node will reboot when CNFS monitoring detects an unrecoverable problem.	yes	no	no	yes	immediately
cnfsSharedRoot Directory to be used by the clustered NFS subsystem	yes	no	yes	no	immediately
cnfsVersions List of protocol versions that CNFS should start and monitor	yes	no	yes	no	immediately

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
dataDiskWaitTimeForRecovery Controls the suspension of dataOnly disk recovery	yes	no	no	yes	immediately
dataStructureDump Path for the storage of dumps	yes	no	no	yes	if not immediately, on restart of the daemon
deadlockBreakupDelay When to attempt breaking up a detected deadlock	yes	yes	no	no	immediately with -i or -I
deadlockDataCollectionDailyLimit Maximum number of times to collect debug data in 24 hours	yes	yes	no	no	immediately with -i or -I
deadlockDataCollectionMinInterval Minimum interval between two consecutive collections of debug data	yes	yes	no	no	immediately with -i or -I
deadlockDetectionThreshold Threshold for detecting deadlocks	yes	yes	no	no	immediately with -i or -I
deadlockDetectionThresholdForShortWaiters Threshold for detecting deadlocks from short waiters	yes	yes	no	no	immediately with -i or -I
deadlockDetectionThresholdIfOverloaded Threshold for detecting deadlocks when a cluster is overloaded	yes	yes	no	no	immediately with -i or -I
deadlockOverloadThreshold Threshold for detecting cluster overload	yes	yes	no	no	immediately with -i or -I
defaultMountDir Default parent directory for GPFS file systems	yes	yes	no	no	for new file systems
disableInodeUpdateOnFdatasync Controls inode update on fdatasync for mtime and atime updates.	yes	yes	no	yes	immediately with -i or -I
dmapiDataEventRetry DMAPI attribute	yes	yes	no	yes	if not immediately, on restart of the daemon
dmapiEventTimeout DMAPI attribute	yes	yes	no	yes	if not immediately, on restart of the daemon

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
dmapiMountEvent DMAPI attribute	yes	yes	no	yes	if not immediately, on restart of the daemon
dmapiMountTimeout DMAPI attribute	yes	yes	no	yes	if not immediately, on restart of the daemon
dmapiSessionFailureTimeout DMAPI attribute	yes	yes	no	yes	if not immediately, on restart of the daemon
enableIPv6 Controls whether the GPFS daemon is to communicate through the IPv6 network.	no	no	only when enableIPv6 is set to yes	not applicable	if not immediately, on restart of the daemon
enforceFilesetQuotaOnRoot Controls fileset quota settings for the root user	yes	yes	no	no	if not immediately, on restart of the daemon
expelDataCollectionDailyLimit Maximum number of times to collect expel-related debug data in 24 hours	yes	yes	no	no	immediately with -i or -I
expelDataCollectionMinInterval Minimum interval between two consecutive collections of expel-related debug data	yes	yes	no	no	immediately with -i or -I
failureDetectionTime Indicates the amount of time it will take to detect that a node has failed	no	no	yes	no	on restart of the daemon
fastestPolicyCmpThreshold Indicates the disk comparison count threshold, above which GPFS forces selection of this disk as the preferred disk to read	yes	yes	no	yes	immediately with -i
fastestPolicyMaxValidPeriod Indicates the time period after which the disk's current evaluation is considered invalid	yes	yes	no	yes	immediately with -i
fastestPolicyMinDiffPercent A percentage value indicating how GPFS selects the fastest between two disks	yes	yes	no	yes	immediately with -i
fastestPolicyNumReadSamples Controls how many read samples taken to evaluate the disk's recent speed	yes	yes	no	yes	immediately with -i

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
fileHeatLossPercent Specifies the reduction rate of FILE_HEAT value for every fileHeatPeriodMinutes of file inactivity.	yes	yes	no	no	if not immediately, on restart of the daemon
fileHeatPeriodMinutes Specifies the inactivity time before a file starts to lose FILE_HEAT value.	yes	yes	no	no	if not immediately, on restart of the daemon
FIPS1402mode Controls whether GPFS operates in FIPS 140-2 mode.	no	no	no	not applicable	on restart of the daemon
forceLogWriteOnFdatasync Controls forcing log writes to disk.	yes	yes	no	yes	immediately with -i or -I
IrocData Controls whether user data will be populated into the local read-only cache.	yes	yes	no	yes	immediately with -i or -I
IrocDataMaxFileSize Limits the data that may be saved in the local read-only cache to only the data from small files.	yes	yes	no	yes	immediately with -i or -I
IrocDataStubFileSize Limits the data that may be saved in the local read-only cache to only the data from the first portion of all files.	yes	yes	no	yes	immediately with -i or -I
IrocDirectories Controls whether directory blocks will be populated into the local read-only cache.	yes	yes	no	yes	immediately with -i or -I
IrocInodes Controls whether inodes from open files will be populated into the local read-only cache.	yes	yes	no	yes	immediately with -i or -I
maxblocksize Maximum file system block size allowed	no	no	no	yes	on restart of the daemon
maxDownDisksForRecovery Maximum number of failed disks allowed for automatic recovery to continue	yes	no	no	yes	immediately
maxFailedNodesForRecovery Maximum number of unavailable nodes allowed before automatic disk recovery is cancelled	yes	no	no	yes	immediately

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of <i>NodeNames</i> allowed	Change takes effect
maxFcntlRangesPerFile Specifies the number of fcntl locks that are allowed per file	yes	yes	no	yes	if not immediately, on restart of the daemon
maxFilesToCache Number of inodes to cache for recently used files	no	no	no	yes	on restart of the daemon
maxMBpS I/O throughput estimate	yes	yes	no	yes	if not immediately, on restart of the daemon
maxStatCache Number of inodes to keep in stat cache	no	no	no	yes	on restart of the daemon
metadataDiskWaitTimeForRecovery Controls the suspension of metadata disk recovery	yes	no	no	yes	immediately
minDiskWaitTimeForRecovery Controls the suspension of disk recovery	yes	no	no	yes	immediately
mmapRangeLock Specifies POSIX or non-POSIX mmap byte-range semantics Note: The list of <i>NodeNames</i> is allowed, but it is not recommended.	yes	yes	no	yes	immediately
nistCompliance Controls whether GPFS operates in NIST 800-131A mode for security transport mechanisms.	no	no	no	not applicable	if not immediately, on restart of the daemon
noSpaceEventInterval Time interval between noDiskSpace events of a file system	yes	yes	no	yes	if not immediately, on restart of the daemon
nsdBufSpace Percentage of the pagepool reserved for the network transfer of NSD requests	yes	yes	no	yes	if not immediately, on restart of the daemon
nsdRAIDBufferPoolSizePct Percentage of the page pool that is used for the GPFS Native RAID vdisk buffer pool	yes	yes	no	yes	if not immediately, on restart of the daemon
nsdRAIDTracks Number of tracks in the GPFS Native RAID buffer pool	yes	yes	no	yes	if not immediately, on restart of the daemon

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
nsdServerWaitTimeForMount Number of seconds to wait for an NSD server to come up	yes	yes	no	yes	if not immediately, on restart of the daemon
nsdServerWaitTimeWindowOnMount Time window to determine if quorum is to be considered <i>recently formed</i>	yes	yes	no	yes	if not immediately, on restart of the daemon
numaMemoryInterleave	no	no	no	yes	on restart of the daemon
pagepool Size of buffer cache on each node	yes	yes	no	yes	if not immediately, on restart of the daemon
pagepoolMaxPhysMemPct Percentage of physical memory that can be assigned to the page pool	no	no	no	yes	on restart of the daemon
pitWorkerThreadsPerNode Maximum number of threads to be involved in parallel processing on each node serving as a Parallel Inode Traversal (PIT) worker	yes	yes	no	yes	immediately with -i or -I
prefetchThreads Maximum number of threads dedicated to prefetching data	no	no	no	yes	on restart of the daemon
readReplicaPolicy The disk read replica policy	yes	yes	no	yes	immediately with -i
release=LATEST Complete the migration to a new release	yes	no	no	no	if not immediately, on restart of the daemon
rpcPerfNumberDayIntervals Number of days that aggregated RPC data is saved	no	no	no	yes	on restart of the daemon
rpcPerfNumberHourIntervals Number of hours that aggregated RPC data is saved	no	no	no	yes	on restart of the daemon
rpcPerfNumberMinuteIntervals Number of minutes that aggregated RPC data is saved	no	no	no	yes	on restart of the daemon
rpcPerfNumberSecondIntervals Number of seconds that aggregated RPC data is saved	no	no	no	yes	on restart of the daemon

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
rpcPerfRawExecBufferSize The buffer size of the raw RPC execution times	no	no	no	yes	on restart of the daemon
rpcPerfRawStatBufferSize The buffer size of the raw RPC statistics	no	no	no	yes	on restart of the daemon
sidAutoMapRangeLength Controls the length of the reserved range for Windows SID to UNIX ID mapping	yes	yes	no	no	if not immediately, on restart of the daemon
sidAutoMapRangeStart Specifies the start of the reserved range for Windows SID to UNIX ID mapping	no	no	no	no	on restart of the daemon
systemLogLevel Filters messages sent to the system log on Linux	yes	yes	no	yes	if not immediately, on restart of the daemon
subnets List of subnets to be used for most efficient daemon-to-daemon communication	no	no	no	yes	on restart of the daemon
tiebreakerDisks (CCR repository) List of tiebreaker disks (NSDs)	no	no	no	no	immediately
tiebreakerDisks (server-based repository) List of tiebreaker disks (NSDs)	no	no	yes	no	on restart of the daemon
uidDomain The UID domain name for the cluster.	no	no	yes	no	on restart of the daemon
unmountOnDiskFail Unmount the file system on a disk failure	yes	yes	no	yes	if not immediately, on restart of the daemon
usePersistentReserve Enables or disables persistent reserve (PR) on the disks	no	no	yes	no	on restart of the daemon
verbsPorts Specifies InfiniBand device names and port numbers	no	no	no	yes	on restart of the daemon
verbsRdma Enables or disables InfiniBand RDMA using the Verbs API.	no	no	no	yes	on restart of the daemon

Table 3. Configuration attributes on the **mmchconfig** command (continued)

Attribute name and Description	-i option allowed	-I option allowed	GPFS must be stopped on all nodes	List of NodeNames allowed	Change takes effect
verbsRdmaCm Enables or disables InfiniBand RDMA_CM using the RDMA_CM API.	no	no	no	yes	on restart of the daemon
verbsRdmaRoCEToS Specifies the Type of Service (ToS) value for clusters using RDMA over Converged Ethernet (RoCE).	yes	yes	no	yes	if not immediately, on restart of the daemon
verbsRdmaSend Enables or disables the use of InfiniBand RDMA rather than TCP for most GPFS daemon-to-daemon communication.	no	no	no	yes	on restart of the daemon
verbsRdmasPerConnection Sets the maximum number of RDMAAs allowed per connection.	yes	yes	no	yes	if not immediately, on restart of the daemon
verbsRdmasPerNode Sets the maximum number of RDMAAs allowed per node.	yes	yes	no	yes	if not immediately, on restart of the daemon
verbsSendBufferMemoryMB Sets the amount of page pool memory to reserve as dedicated buffer space for use by verbsRdmaSend .	yes	yes	no	yes	if not immediately, on restart of the daemon
workerThreads Sets an integrated group of variables that tune file system performance.	no	no	no	yes	on restart of the daemon
worker1Threads Sets the maximum number of concurrent file operations	yes (only when adjusting value down)	yes (only when adjusting value down)	no	yes	on restart of the daemon

Specify the nodes you want to target for change and the attributes with their new values on the **mmchconfig** command. For example, to change the **pagepool** value for each node in the GPFS cluster immediately, enter:

```
mmchconfig pagepool=100M -i
```

The system displays information similar to:

```
mmchconfig: Command successfully completed
mmchconfig: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

For complete usage information, see “mmchconfig command” on page 331.

Security mode

The security mode of a cluster determines the level of security that the cluster provides for communications between nodes in the cluster and also for communications between clusters.

There are three security modes:

EMPTY

The receiving node and the sending node do not authenticate each other, do not encrypt transmitted data, and do not check the integrity of transmitted data.

AUTHONLY

The sending and receiving nodes authenticate each other with a TLS handshake and then close the TLS connection. Communication continues in the clear. The nodes do not encrypt transmitted data and do not check data integrity.

Cipher To set this mode, you must specify the name of a supported cipher, such as AES128-GCM-SHA256. The sending and receiving nodes authenticate each other with a TLS handshake. A TLS connection is established. The transmitted data is encrypted with the specified cipher and is checked for data integrity.

To find a list of supported ciphers, choose one of the following methods:

- See the frequently answered questions (FAQs) in IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- Enter the following command at the command line:

```
mmauth show ciphers
```

For FIPS 140-2 considerations, see the *Encryption* topic in the *IBM Spectrum Scale: Advanced Administration Guide*.

For both the **AUTHONLY** mode and the cipher mode, the cluster automatically generates a public/private key pair when the mode is set. However, for communication between clusters, the system administrators are still responsible for exchanging public keys.

In IBM Spectrum Scale V4.2 or later, the default security mode is **AUTHONLY**. The **mmcrcluster** command sets the mode when it creates the cluster. You can display the security mode by running the following command:

```
mmisconfig cipherlist
```

You can change the security mode with the following command:

```
mmchconfig cipherlist=security_mode
```

If you are changing the security mode from **EMPTY** to another mode, you can do so without stopping the GPFS daemon. However, if you are changing the security mode from another mode to **EMPTY**, you must stop the GPFS daemon on all the nodes in the cluster. Change the security mode to **EMPTY** and then restart the GPFS daemons.

In IBM Spectrum Scale V4.1 or earlier, the default security mode is **EMPTY**. If you update a cluster from IBM Spectrum Scale V4.1 to V4.2 or later by running `mmchconfig release=LATEST`, the command checks the security mode. If the mode is **EMPTY**, the command issues a warning message but does not change the security mode of the cluster.

Configuring the security mode to a setting other than **EMPTY** (that is, either **AUTHONLY** or a supported cipher) requires the use of the GSKit toolkit for encryption and authentication. As such, the **gpfs.gskit** package, which is available on all Editions, should be installed.

Related reference:

“mmauth command” on page 276
Manages secure access to GPFS file systems.
“mmchconfig command” on page 331
Changes GPFS configuration parameters.
“mmlsconfig command” on page 526
Displays the current configuration data for a GPFS cluster.

Running IBM Spectrum Scale without remote root login

You can avoid configuring your GPFS nodes to allow remote login to the root user ID, by using sudo wrapper scripts to run GPFS administration commands.

Every administration node in the IBM Spectrum Scale cluster must be able to run the administration commands, generally known as *mm commands*, on any other node in the cluster. Each administration node must be able to do so without the use of a password and without producing any extraneous messages. Also, most of the IBM Spectrum Scale administration commands must run at the root level. One solution to meet these requirements is to configure each node to permit general remote login to its root user ID. However, there are secure solutions available that do not require root-level login.

You can use a **sudo** program, or a sudo-like framework to enable a user login, which is not at the root-level. With sudo wrapper, you can launch IBM Spectrum Scale administration commands with a sudo wrapper script. This script uses **ssh** to log in to the remote node using a non-root ID, and then use sudo on the remote node to run the commands with root-level privileges. The root user on an administration node still needs to be able to log in to all nodes in the cluster as the non-root ID, without being prompted for a password.

Note: Sudo wrappers are not supported on clusters where one or more of the nodes is running the Windows operating system.

To use sudo wrappers, complete the tasks in the following links:

Configuring sudo

The system administrator must configure sudo by modifying the sudoers file. IBM Spectrum Scale installs a sample of the modified sudoers file as `/usr/lpp/mmfs/samples/sudoers.sample`.

Perform the following steps before you start configuring sudo:

1. Create a user and group to run administration commands.
The examples in this section use the user name as `gpfsadmin` and the group `asgpfs`.
2. Allow password-less access to root user from any administration node to issue commands on all nodes with the user `IDgpfsadmin`.
3. Install the sudo program. Sudo is a free open source software that is distributed under a license.

Do the following steps on each node in the cluster:

1. Open the `/etc/sudoers` file with a text editor. The sudo installation includes the *visudo* editor, which checks the syntax of the file before closing.
2. Add the following commands to the file. **Important:** Enter each command on a single line:

```
# Preserve GPFS environment variables:
Defaults env_keep += "MMMODE environmentType GPFS_rshPath GPFS_rcpPath mmScriptTrace GPFS_CMDPORTRANGE GPFS_CIM_MSG_FORMAT"

# Allow members of the gpfs group to run all commands but only selected commands without a password:
%gpfs ALL=(ALL) PASSWD: ALL, NOPASSWD: /usr/lpp/mmfs/bin/mmremote, /usr/bin/scp, /bin/echo, /usr/lpp/mmfs/bin/mmsdrrestore

# Disable requiretty for group gpfs:
Defaults:%gpfs !requiretty
```

The first line preserves the environment variables that the IBM Spectrum Scale administration commands need to run. The second line allows the users in the `gpfs` group to run administration commands without being prompted for a password. The third line disables `requiretty`. When this flag is enabled, `sudo` blocks the commands that do not originate from a TTY session.

3. Perform the following steps to verify that the `sshwrap` and `scpwrap` scripts work correctly.
 - a. `sshwrap` is an IBM Spectrum Scale `sudo` wrapper script for the remote shell command that is installed with IBM Spectrum Scale. To verify that it works correctly, run the following command as the `gpfsadmin` user:

```
sudo /usr/lpp/mmfs/bin/mmcommon test sshwrap nodeName
[sudo] password for gpfsadmin:
mmcommon test sshwrap: Command successfully completed
```

Note: `nodeName` is the name of an IBM Spectrum Scale node in the cluster

- b. `scpwrap` is an IBM Spectrum Scale `sudo` wrapper script for the remote file copy command that is installed with IBM Spectrum Scale. To verify that it works correctly, run the following command as the `gpfsadmin` user:

```
sudo /usr/lpp/mmfs/bin/mmcommon test scpwrap nodeName
mmcommon test scpwrap: Command successfully completed
```

Note: `nodeName` is the name of an IBM Spectrum Scale node in the cluster

`Sudo` is now configured to run administration commands without remote root login.

Configuring the cluster to use `sudo` wrapper scripts

The system administrator must configure the IBM Spectrum Scale cluster to call the `sudo` wrapper scripts `sshwrap` and `scpwrap` to run IBM Spectrum Scale administration commands. To configure the cluster, run either the `mmcrcluster` command or the `mmchcluster` command with the `--use-sudo-wrapper` option.

Perform the following steps to configure a new cluster or an existing to call the `sudo` wrapper scripts:

- To configure a new cluster to call the `sudo` wrapper scripts, use these steps.
 1. Log in with the user ID. This example uses `gpfsadmin` as the user ID.
 2. Issue the `mmcrcluster` command with the `--use-sudo-wrapper` option as shown in the following example:

```
$ sudo /usr/lpp/mmfs/bin/mmcrcluster --use-sudo-wrapper -Nc13c1apv7:quorum,c13c1apv8
mmcrcluster: Performing preliminary node verification ...
mmcrcluster: Processing quorum and other critical nodes ...
mmcrcluster: Processing the rest of the nodes ...
mmcrcluster: Finalizing the cluster data structures ...
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper GPFS license designations.
mmcrcluster: Use the mmchlicense command to designate licenses as needed.
mmcrcluster: Propagating the cluster configuration data to all
mmcrcluster: affected nodes. This is an asynchronous process
```

3. To verify that the cluster is using `sudo` wrappers, issue the `mmclscluster` command as shown in the following example:

```
gpfsadmin@c13c1apv7 admin]$mmclscluster
GPFS cluster information
=====
GPFS cluster name: c13c1apv7.gpfs.net
GPFS cluster id: 12275146245716580740
GPFS UID domain: c13c1apv7.gpfs.net
Remote shell command: sudo wrapper in use
Remote file copy command: sudo wrapper in use
Repository type: CCR
Node Daemon node name IP address Admin node name Designation
```

```
-----  
1 c13c1apv7.gpfs.net 192.168.148.117 c13c1apv7.gpfs.net quorum  
2 c13c1apv8.gpfs.net 192.168.148.118 c13c1apv8.gpfs.net
```

- To configure an existing cluster to call the sudo wrapper scripts, use these steps.
 1. Log in with the user ID. This example uses gpfsadmin as the user ID.
 2. Issue the **mmchcluster** command with the **--use-sudo-wrapper** option to start using the sudo wrappers:

```
sudo /usr/lpp/mmfs/bin/mmchcluster --use-sudo-wrapper
```
 3. To verify that cluster is using sudo wrappers, issue the **mmlscluster** command with no parameters. If sudo wrapper is configured properly, the output must contain the following two lines.

```
Remote shell command: sudo wrapper in use  
Remote file copy command: sudo wrapper in use
```

Configuring a cluster to stop using sudo wrapper scripts

You can opt to stop using sudo wrapper scripts in the IBM Spectrum Scale cluster.

To stop using sudo wrappers, run the **mmchcluster** command with the **--nouse-sudo-wrapper** option as shown in the following example:

```
$sudo /usr/lpp/mmfs/bin/mmchcluster --nouse-sudo-wrapper
```

Now, the cluster stops calling the sudo wrapper scripts to run the remote administration commands.

Node quorum considerations

A node quorum is the minimum number of nodes that must be running in order for the daemon to start. Node quorum is the default quorum algorithm for GPFS.

For more information on node quorum, see the section on *Quorum*, in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*

Node quorum with tiebreaker considerations

Node quorum with tiebreaker disks allows you to run with as little as one quorum node available as long as you have access to a majority of the quorum disks. Enabling node quorum with tiebreaker disks starts by designating one or more nodes as quorum nodes. Then one to three disks are defined as tiebreaker disks using the **tiebreakerDisks** parameter on the **mmchconfig** command. You can designate any disk to be a tiebreaker. When utilizing node quorum with tiebreaker disks, there are specific rules for cluster nodes and for tiebreaker disks.

For more information on node quorum with tiebreaker, see the section on *Quorum*, in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*

When using node quorum with tiebreaker, define one, two, or three disks to be used as tiebreaker disks when any quorum node is down. Issue this command:

```
mmchconfig tiebreakerDisks="nsdName;nsdName;nsdName"
```

Consider these points:

- You are not permitted to change a GPFS cluster configuration to use node quorum with tiebreaker if there are more than eight existing quorum nodes.
- You can have a maximum of three tiebreaker disks.
- The disks must be directly attached to all quorum nodes.
- When adding tiebreaker disks:
 - If the tiebreaker disks are part of a file system, GPFS should be up and running.

- If the tiebreaker disks are not part of a file system, GPFS can be either running or shut down.
- When using the traditional server-based (non-CCR) configuration repository, the GPFS daemons must be down on all nodes in the cluster when running **mmchconfig tiebreakerDisks**.

If you are using node quorum with tiebreaker and want to change to using node quorum, issue this command:

```
mmchconfig tiebreakerDisks=DEFAULT
```

Displaying and changing the file system manager node

In general, GPFS performs the same functions on all nodes. There are also cases where one node provides a more global function that affects the operation of multiple nodes. For example, each file system is assigned a node that functions as a file system manager.

For a more detailed discussion on the role of the file system manager node, see *Special management functions* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

The node that is the file system manager can also be used for applications. In some cases involving very large clusters or applications that place a high stress on metadata operations, it may be useful to specify which nodes are used as file system managers. Applications that place a high stress on metadata operations are usually those that involve large numbers of very small files, or that do very fine-grain parallel write-sharing among multiple nodes.

You can display the file system manager node by issuing the **mmlsmgr** command. You can display the information for an individual file system, a list of file systems, or for all of the file systems in the cluster. For example, to display the file system manager for the file system **fs1**, enter:

```
mmlsmgr fs1
```

The output shows the device name of the file system and the file system manager's node number and name:

```
file system      manager node      [from 19.134.68.69 (k164n05)]
-----
fs1              19.134.68.70 (k164n06)
```

For complete usage information, see “mmlsmgr command” on page 542.

You can change the file system manager node for an individual file system by issuing the **mmchmgr** command. For example, to change the file system manager node for the file system **fs1** to **k145n32**, enter:

```
mmchmgr fs1 k145n32
```

The output shows the file system manager's node number and name, in parentheses, as recorded in the GPFS cluster data:

```
GPFS: 6027-628 Sending migrate request to current manager node 19.134.68.69 (k145n30).
GPFS: 6027-629 [N] Node 19.134.68.69 (k145n30) resigned as manager for fs1.
GPFS: 6027-630 [N] Node 19.134.68.70 (k145n32) appointed as manager for fs1.
```

For complete usage information, see “mmchmgr command” on page 379.

Determining how long mmrestripefs takes to complete

There are several factors that determine how long the **mmrestripefs** command takes to complete.

To determine how long the **mmrestripefs** command will take to complete, consider these points:

1. How much data potentially needs to be moved. You can estimate this using the **df** command.
2. How many GPFS client nodes there are to do the work.

3. How much Network Shared Disk (NSD) server bandwidth is available for I/O.
4. If you have added new disks to a file system, use the **mmddf** command to determine how much free space is available on each of the new disks.

The restriping of a file system is done by having multiple threads on each node in the cluster work on a subset of files. If the files are large, multiple nodes can participate in restriping it in parallel. Consequently, the more GPFS client nodes there are performing work for the restripe, the faster the **mmrestripefs** command will complete. The nodes that should participate in the restripe are specified on the command using the **-N** parameter. Based on raw I/O rates, you should be able to estimate the length of time for the restripe. However, to account for the overhead of scanning all metadata, that value should be doubled.

Assuming that you have enough nodes to saturate the disk servers, and have to move all of the data, the time to read and write every block of data is roughly:

$$2 * \text{fileSystemSize} / \text{averageDiskserverDataRate}$$

As an upper bound, due to overhead of scanning all of the metadata, this time should be doubled. If other jobs are loading the NSD servers heavily, this time may increase even more.

Note: There is no particular reason to stop all other jobs while the **mmrestripefs** command is running. The CPU load of the command is minimal on each node and only the files that are being restriped at any moment are locked to maintain data integrity.

Starting and stopping GPFS

You can use the **mmstartup** and **mmshutdown** commands to start and stop GPFS on new or existing clusters.

For new GPFS clusters, see *Steps to establishing and starting your GPFS cluster* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For existing GPFS clusters, before starting GPFS, ensure that you have:

1. Verified the installation of all prerequisite software.
2. Compiled the GPL layer, if Linux is being used.
3. Properly configured and tuned your system for use by GPFS. This should be done prior to starting GPFS.

For details, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Start the daemons on all of the nodes in the cluster by issuing the **mmstartup -a** command:

```
mmstartup -a
```

The output is similar to this:

```
Tue Aug 24 15:54:56 edt 2004: 6027-1642 mmstartup: Starting GPFS ...
```

Check the messages recorded in **/var/adm/ras/mmfs.log.latest** on one node for verification. Look for messages similar to this:

```
GPFS: 6027-300 [N] mmfsd ready
```

This indicates that quorum has been formed and this node has successfully joined the cluster, and is now ready to mount file systems.

If GPFS does not start, see *GPFS daemon will not come up* in *IBM Spectrum Scale: Problem Determination Guide*.

For complete usage information, see “mmstartup command” on page 678.

If it becomes necessary to stop GPFS, you can do so from the command line by issuing the **mmshutdown** command:

```
mmshutdown -a
```

The system displays information similar to:

```
Thu Aug 12 13:10:40 EDT 2004: 6027-1341 mmshutdown: Starting force unmount of GPFS file systems
k164n05.kgn.ibm.com: forced unmount of /fs1
k164n04.kgn.ibm.com: forced unmount of /fs1
k164n06.kgn.ibm.com: forced unmount of /fs1
Thu Aug 12 13:10:45 EDT 2004: 6027-1344 mmshutdown: Shutting down GPFS daemons
k164n04.kgn.ibm.com: Shutting down!
k164n06.kgn.ibm.com: Shutting down!
k164n05.kgn.ibm.com: Shutting down!
k164n04.kgn.ibm.com: 'shutdown' command about to kill process 49682
k164n05.kgn.ibm.com: 'shutdown' command about to kill process 28194
k164n06.kgn.ibm.com: 'shutdown' command about to kill process 30782
Thu Aug 12 13:10:54 EDT 2004: 6027-1345 mmshutdown: Finished
```

For complete usage information, see “mmshutdown command” on page 661.

Chapter 3. Managing file systems

There are several file system management tasks outlined in this topic.

If you need information about how to create GPFS file systems, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the `mmcrfs` command.

File system management tasks include:

1. "Mounting a file system"
2. "Unmounting a file system" on page 29
3. "Deleting a file system" on page 29
4. "Determining which nodes have a file system mounted" on page 30
5. "Checking and repairing a file system" on page 30
6. "Listing file system attributes" on page 33
7. "Modifying file system attributes" on page 34
8. "Querying and changing file replication attributes" on page 34
9. "Using Direct I/O on a file in a GPFS file system" on page 35
10. "Restripping a GPFS file system" on page 42
11. "Querying file system space" on page 43
12. "Querying and reducing file system fragmentation" on page 44
13. "Protecting data in a file system using backup" on page 46
14. "Scale Out Backup and Restore (SOBAR)" on page 53

Managing filesets, storage pools and policies is also a file system management task. See the *IBM Spectrum Scale: Advanced Administration Guide* for more information. Use the following information to manage file systems in IBM Spectrum Scale.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Files > File Systems**.

Note: Creation of file systems is not supported in the 4.2 GUI.

Mounting a file system

You must explicitly mount a GPFS file system if this is the first time the file system is being mounted after its creation, or you specified *not to* automatically mount (`-A no`) the file system when you created it.

If you allowed the default value for the automatic mount option (`-A yes`) when you created the file system, then you do not need to use this procedure after restarting GPFS on the nodes.

To mount a GPFS file system, enter:

```
mmmount device
```

where *device* is the name of the file system. For example, to mount the file system `fs1`, enter:

```
mmmount fs1
```

Mounting a file system on multiple nodes

This topic describes how to mount a file systems on multiple nodes.

To mount file system **fs1** on all nodes in the GPFS cluster, issue this command:

```
mmmount fs1 -a
```

To mount a file system only on a specific set of nodes, use the **-N** flag of the **mmmount** command.

GPFS-specific mount options

GPFS-specific mount options can be specified with the **-o** parameter on the **mmchfs**, **mmremotefs**, **mmmount** and **mount** commands. Options specified with the **mmchfs** and **mmremotefs** commands are recorded in the GPFS configuration files and are passed as default options to subsequent mount commands on all nodes in the cluster. Options specified with the **mmmount** or **mount** commands override the existing default settings, and are not persistent.

All of the mount options can be specified using the **-o** parameter. Multiple options should be separated only by a comma. If an option is specified multiple times, the last instance is the one that takes effect. Certain options can also be set with specifically designated command flags. Unless otherwise stated, mount options can be specified as:

option or *option=1* or *option=yes* - to enable the option

nooption or *option=0* or *option=no* - to disable the option

The *option={1 | 0 | yes | no}* syntax should be used for options that can be intercepted by the **mount** command and not passed through to GPFS. An example is the **atime** option in the Linux environment.

The GPFS-specific mount options are:

atime

Update inode access time for each access. This is the default. This option can also be controlled with the **-S** option on the **mmcrfs** and **mmchfs** commands.

mtime

Always return accurate file modification times. This is the default. This option can also be controlled with the **-E** option on the **mmcrfs** and **mmchfs** commands.

noatime

Do not update inode access times on this file system. This option can also be controlled with the **-S** option on the **mmcrfs** and **mmchfs** commands.

nomtime

Update file modification times only periodically. This option can also be controlled with the **-E** option on the **mmcrfs** and **mmchfs** commands.

norelatime

Update inode access time for each access. This is the default.

This option can also be controlled with the **-S** option on the **mmcrfs** and **mmchfs** commands.

nosyncnfs

Do not commit metadata changes coming from the NFS daemon synchronously. Normal file system synchronization semantics apply. This is the default.

relatime

Allow the update of inode access time only if either of the following is true:

- The existing access time is older than 24 hours. (Access time is user configurable through the **atimeDeferredSeconds** configuration attribute.)

- The existing file modification time is greater than the existing access time.

This option can also be controlled with the **-S** option on the **mmcrfs** and **mmchfs** commands.

syncnfs

Synchronously commit metadata changes coming from the NFS daemon.

useNSDserver={always | asfound | asneeded | never}

Controls the initial disk discovery and failover semantics for NSD disks. The possible values are:

always

Always access the disk using the NSD server. Local dynamic disk discovery is disabled.

asfound

Access the disk as found (the first time the disk was accessed). No change of disk access from local to NSD server, or the other way around, is performed by GPFS.

asneeded

Access the disk any way possible. This is the default.

never

Always use local disk access.

Unmounting a file system

Some GPFS administration tasks require you to unmount the file system before they can be performed. You can unmount a GPFS file system using the **mmumount** command.

If the file system will not unmount, see the *IBM Spectrum Scale: Problem Determination Guide* and search for *file system will not unmount*.

To unmount a GPFS file system using the **mmumount** command, enter:

```
mmumount device
```

where *device* is the name of the file system. For example, to unmount the file system **fs1**, enter:

```
mmumount fs1
```

Unmounting a file system on multiple nodes

This topic describes how to unmount a file systems on multiple nodes.

To unmount file system **fs1** on all nodes in the GPFS cluster, issue this command:

```
mmumount fs1 -a
```

To unmount a file system only on a specific set of nodes, use the **-N** flag of the **mmumount** command.

Deleting a file system

Before deleting a file system, unmount it on all nodes.

Specify the file system to be deleted on the **mmdelfs** command. For example, to delete the file system **fs1**, enter:

```
mmdelfs fs1
```

The system displays information similar to:

GPFS: 6027-573 All data on the following disks of fs1 will be destroyed:

```
gpfs9nsd
gpfs13nsd
gpfs11nsd
```

gpfs12nsd

GPFS: 6027-574 Completed deletion of file system fs1.
mmdelfs: 6027-1371 Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

For more information, see the following:

- “Unmounting a file system” on page 29
- “mmdelfs command” on page 458 for complete usage information
- “mmdelnsd command” on page 465 for removing the NSD definitions after deleting the file system

Determining which nodes have a file system mounted

The **mmlsmount** command is used to determine which nodes have a given file system mounted. The name and IP address of each node that has the file system mounted is displayed. This command can be used for all file systems, all remotely mounted file systems, or file systems mounted on nodes of certain clusters.

Note that the **mmlsmount -L** command reports file systems that are in use at the time the command is issued. A file system is considered to be in use if it is explicitly mounted with the **mount** or **mmmount** command or if it is mounted internally for the purposes of running some other GPFS command. For example, when you run the **mmrestripefs** command, the file system will be internally mounted for the duration of the command. If **mmlsmount** is issued in the interim, the file system will be reported as being in use by the **mmlsmount** command but, unless it is explicitly mounted, will not show up in the output of the **mount** or **df** commands.

This is an example of a **mmlsmount -L** command for a mounted file system named **fs1**:

```
File system fs1 (mnsd.cluster:fs1) is mounted on 5 nodes:
 9.114.132.101 c5n101          mnsd.cluster
 9.114.132.100 c5n100          mnsd.cluster
 9.114.132.106 c5n106          mnsd.cluster
 9.114.132.97  c5n97           cluster1.cluster
 9.114.132.92  c5n92           cluster1.cluster
```

Checking and repairing a file system

The **mmfsck** command finds and repairs conditions that can cause problems in your file system. The **mmfsck** command operates in two modes: online and offline.

The online mode operates on a mounted file system and is chosen by issuing the **-o** option. Conversely, the offline mode operates on an unmounted file system. In general it is unnecessary to run **mmfsck** in offline mode unless under the direction of the IBM Support Center.

The online mode only checks and recovers unallocated blocks on a mounted file system. If a GPFS file operation fails due to an out of space condition, the cause may be disk blocks that have become unavailable after repeated node failures. The corrective action taken is to mark the block free in the allocation map. Any other inconsistencies found are only reported, not repaired.

Note:

1. If you are running the online **mmfsck** command to free allocated blocks that do not belong to any files, plan to make file system repairs when system demand is low. This is I/O intensive activity and it can affect system performance.
2. If you are repairing a file system due to node failure and the file system has quotas enabled, it is suggested that you run the **mmcheckquota** command to recreate the quota files.

To repair any other inconsistencies, you must run the offline mode of the **mmfsck** command on an unmounted file system. The offline mode checks for these file inconsistencies that might cause problems:

- Blocks marked allocated that do not belong to any file. The corrective action is to mark the block free in the allocation map.
- Files and directories for which an inode is allocated and no directory entry exists, known as orphaned files. The corrective action is to create directory entries for these files in a **lost+found** subdirectory in the root directory of the fileset to which the file or directory belongs. A fileset is a subtree of a file system namespace that in many respects behaves like an independent file system. The index number of the inode is assigned as the name. If you do not allow the **mmfsck** command to reattach an orphaned file, it asks for permission to delete the file.
- Directory entries pointing to an inode that is not allocated. The corrective action is to remove the directory entry.
- Incorrectly formed directory entries. A directory file contains the inode number and the generation number of the file to which it refers. When the generation number in the directory does not match the generation number stored in the file's inode, the corrective action is to remove the directory entry.
- Incorrect link counts on files and directories. The corrective action is to update them with accurate counts.
- Policy files that are not valid. The corrective action is to delete the file.
- Various problems related to filesets: missing or corrupted fileset metadata, inconsistencies in directory structure related to filesets, missing or corrupted fileset root directory, other problems in internal data structures. The repaired filesets will be renamed as **FilesetFilesetId** and put into unlinked state.

The **mmfsck** command performs other functions not listed here, as deemed necessary by GPFS.

The **--patch-file** parameter of the **mmfsck** command can be used to generate a report of file system inconsistencies. The following is an example of a patch file that is generated by **mmfsck** for a file system with a bad directory inode:

```
gpfs_fsck
```

```
<header>
  sgid = "C0A87ADC:5555C87F"
  disk_data_version = 1
  fs_name = "gpfs0"
  #patch_file_version = 1
  #start_time = "Fri May 15 16:32:58 2015"
  #fs_manager_node = "h0"
  #fsck_flags = 150994957
</header>
```

```
<patch_inode>
  patch_type = "dealloc"
  snapshot_id = 0
  inode_number = 50432
</patch_inode>
```

```
<patch_block>
  snapshot_id = 0
  inode_number = 3
  block_num = 0
  indirection_level = 0
  generation_number = 1
  is_clone = false
  is_directory_block = true
  rebuild_block = false
  #num_patches = 1
```

```
<patch_dir>
  entry_offset = 48
  entry_fold_value = 306661480
```

```

    delete_entry = true
  </patch_dir>
</patch_block>

<patch_block>
  snapshot_id = 0
  inode_number = 0
  block_num = 0
  indirection_level = 0
  generation_number = 4294967295
  is_clone = false
  is_directory_block = false
  rebuild_block = false
  #num_patches = 1

  <patch_field>
    record_number = 3
    field_id = "inode_num_links"
    new_value = 2
    old_value = 3
  </patch_field>
</patch_block>

<patch_inode>
  patch_type = "orphan"
  snapshot_id = 0
  inode_number = 50433
</patch_inode>

<footer>
  #stop_time = "Fri May 15 16:33:06 2015"
  #num_sections = 203
  #fsck_exit_status = 8
  need_full_fsck_scan = false
</footer>

```

The **mmfsck** command can be run with both the **--patch-file** and **--patch** parameters to repair a file system with the information stored in the patch file. Using a patch file prevents a subsequent scan of the file system before the repair actions begin.

You cannot run the **mmfsck** command on a file system that has disks in a **down** state. You must first run the **mmchdisk** command to change the state of the disks to **unrecovered** or **up**. To display the status of the disks in the file system, issue the **mmlsdisk** command.

To check the file system **fs1** without making any changes to the file system, issue the following command:

```
mmfsck fs1
```

For complete usage information, see “mmchdisk command” on page 354, “mmcheckquota command” on page 361, “mmfsck command” on page 487, and “mmlsdisk command” on page 528.

Dynamic validation of descriptors on disk

GPFS has the ability to periodically scan descriptors on disk to detect and fix corruption early rather than waiting until the next remount.

The first time a file system gets mounted, a periodic validation of the nsd, disk, and stripe group descriptors gets started. This validation occurs, by default, every five seconds. The nsd, disk, and stripe group descriptors are read and compared with the corresponding descriptors in memory/cache. If there is a mismatch, that information is logged and, if appropriate, the corrupted data is fixed using data from cache.

Listing file system attributes

Use the **mmlsfs** command to display the current file system attributes. Depending on your configuration, additional information which is set by GPFS may be displayed to assist in problem determination when contacting the IBM Support Center.

If you specify no options with the **mmlsfs** command, all file system attributes are listed.

For example, to list all of the attributes for the file system **gpfs1**, enter:

```
mmlsfs gpfs1
```

The system displays information similar to:

flag	value	description
-f	8192	Minimum fragment size in bytes
-i	4096	Inode size in bytes
-I	16384	Indirect block size in bytes
-m	2	Default number of metadata replicas
-M	2	Maximum number of metadata replicas
-r	2	Default number of data replicas
-R	2	Maximum number of data replicas
-j	cluster	Block allocation type
-D	nfs4	File locking semantics in effect
-k	all	ACL semantics in effect
-n	32	Estimated number of nodes that will mount file system
-B	262144	Block size
-Q	user;group;fileset	Quotas accounting enabled
	user;group;fileset	Quotas enforced
	none	Default quotas enabled
--perfileset-quota	no	Per-fileset quota enforcement
--filesetdf	no	Fileset df enabled?
-V	14.20 (4.1.1.0)	File system version
--create-time	Fri Jun 12 18:39:47 2015	File system creation time
-z	no	Is DMAPI enabled?
-L	134217728	Logfile size
-E	yes	Exact mtime mount option
-S	no	Suppress atime mount option
-K	whenpossible	Strict replica allocation option
--fastea	yes	Fast external attributes enabled?
--encryption	no	Encryption enabled?
--inode-limit	607488	Maximum number of inodes in all inode spaces
--log-replicas	2	Number of log replicas
--is4KAligned	yes	is4KAligned?
--rapid-repair	yes	rapidRepair enabled?
--write-cache-threshold	65536	HAWC Threshold (max 65536)
-P	system	Disk storage pools in file system
-d	nsd20;nsd21;nsd3	Disks in file system
-A	yes	Automatic mount option
-o	none	Additional mount options
-T	/gpfs1	Default mount point
--mount-priority	0	Mount priority

Note that some of the attributes displayed by the **mmlsfs** command represent default mount options. Since the scope of mount options is an individual node, it is possible to have different values on different nodes. For exact **mtime** (-E option) and suppressed **atime** (-S option), the information displayed by the **mmlsfs** command represents the current setting on the file system manager node. If these options are changed with the **mmchfs** command, the change may not be reflected until the file system is remounted.

For complete usage information, see “mmlsfs command” on page 536. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on *GPFS architecture* and *file system creation considerations* for a detailed discussion of file system attributes.

Modifying file system attributes

Use the **mmchfs** command to modify existing file system attributes.

Note: All files created after issuing the **mmchfs** command take on the new attributes. Existing files are not affected. Use the **mmchattr** or **mmrestripefs -R** command to change the replication factor of existing files. See “Querying and changing file replication attributes.”

For example, to change the default data replication factor to 2 for the file system **fs1**, enter:

```
mmchfs fs1 -r 2
```

To confirm the changes, enter:

```
mmlsfs fs1 -r
```

The system displays information similar to:

flag	value	description
-r	2	Default number of data replicas

For complete usage information, see “mmchfs command” on page 371 and “mmlsfs command” on page 536. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on *GPFS architecture* and *file system creation considerations* for a detailed discussion of file system attributes.

Querying and changing file replication attributes

If your availability requirements change, you can have GPFS display the current replication factors for one or more files by issuing the **mmlsattr** command. You might then decide to change replication for one or more files using the **mmchattr** command.

For complete usage information, see “mmlsattr command” on page 519 and “mmchattr command” on page 321.

Querying file replication

Specify one or more file names with the **mmlsattr** command.

For example, to display the replication factors for two files named **project4.sched** and **project4.resource** in the file system **fs1**, enter:

```
mmlsattr /fs1/project4.sched /fs1/project4.resource
```

The system displays information similar to:

replication factors		
metadata(max)	data(max)	file [flags]
1 (2)	1 (2)	/fs1/project4.sched
1 (2)	1 (2)	/fs1/project4.resource

See the “mmlsattr command” on page 519 for complete usage information. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on *GPFS architecture* and *File system creation considerations* for a detailed discussion of file system attributes.

Changing file replication attributes

Use the **mmchattr** command to change the replication attributes for one or more files.

You can only increase data and metadata replication as high as the maximum data and maximum metadata replication factors for that file system. You cannot change the maximum data and maximum metadata replication factors once the file system has been created.

Specify the file name, attribute, and new value with the **mmchattr** command. For example, to change the metadata replication factor to 2 and the data replication factor to 2 for the file named **project7.resource** in the file system **fs1**, enter:

```
mmchattr -m 2 -r 2 /fs1/project7.resource
```

To confirm the change, enter:

```
mmisattr /fs1/project7.resource
```

The system displays information similar to:

```
replication factors
metadata(max) data(max) file   [flags]
-----
      2 ( 2)   2 ( 2) /fs1/project7.resource
```

See the “**mmchattr** command” on page 321 and the “**mmisattr** command” on page 519 for complete usage information. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on *GPFS architecture* and *File system creation considerations* for a detailed discussion of file system attributes.

Using Direct I/O on a file in a GPFS file system

The Direct I/O caching policy can be set for files in a GPFS file system by specifying the **-D** option on the **mmchattr** command.

This caching policy bypasses file cache and transfers data directly from disk into the user space buffer, as opposed to using the normal cache policy of placing pages in kernel memory. Applications with poor cache hit rates or very large I/Os may benefit from the use of Direct I/O.

Direct I/O may also be specified by supplying the **O_DIRECT** file access mode on the **open()** of the file.

File compression

You can compress or decompress files either with the **mmchattr** command or with the **mmapplypolicy** command with a **MIGRATE** rule. You can do the compression or decompression synchronously or defer it until a later call to **mmrestripefile** or **mmrestripefs**.

| IBM Spectrum Scale V4.2 adds file compression to reduce the size of data at rest. File compression is
| intended primarily for cold data and favors saving space over access speed. File compression can be
| driven by policies that enabled administrators to compress only files that are not accessed for some
| specified time. Data is decompressed inline for each read access.

For more information about file compression, see the following subtopics:

- | • “Comparison with object compression” on page 36
- | • “When to use file compression” on page 36
- | • “Setting up file compression and decompression” on page 36
- | • “Warnings” on page 37
- | • “Reported size of compressed files” on page 37
- | • “Deferred file compression” on page 37
- | • “Indicators of file compression or decompression” on page 37
- | • “Updates to compressed files” on page 39
- | • “File compression and memory mapping” on page 39

- “File compression and direct I/O” on page 39
- “Backing up and restoring compressed files” on page 39
- “Limitations” on page 39

Comparison with object compression

File compression is a different feature than object compression. Both features compress files, and both features can be policy-driven. However, object compression is available only through Cluster Export Systems (CES) and is done with the **mmobj** command. File compression is available outside CES and is done with the **mmapplypolicy** command (policy-driven) or the **mmchattr** command (direct). Also, with file compression you can defer the compression or decompression operation until a time when the system is not loaded with processes and I/O. For more information about object compression, see the topic “Administering storage policies for object storage” on page 131.

When to use file compression

File compression in this release is designed to be used only for compressing cold data or write-once objects and files. Compressing other types of data can result in performance degradation. File compression uses the zlib data compression library and favors saving space over speed.

Setting up file compression and decompression

The sample script `/usr/lpp/mmfs/samples/ilm/mmcompress.sample`, installed with IBM Spectrum Scale, provides examples of how to compress or decompress a fileset or a directory tree.

You can do file compression or decompression with either the **mmchattr** command or the **mmapplypolicy** command.

Note: File compression and decompression with the **mmapplypolicy** command is not supported on Windows.

With the **mmchattr** command, you specify the **-compression** option and the names of the files or filesets that you want to compress or decompress. For example, the following command compresses a file:

```
mmchattr --compression yes trcrpt.150913.13.30.13.3518.txt
```

The following command decompresses the same file:

```
mmchattr --compression no trcrpt.150913.13.30.13.3518.txt
```

For more information, see the topic *mmchattr command* in the *IBM Spectrum Scale: Administration and Programming Reference*.

With the **mmapplypolicy** command, you create a **MIGRATE** rule that specifies the **COMPRESS** option and run **mmapplypolicy** to apply the rule. For example, the following rule, which applies to files with names that contain the string `green`, migrates files out of a storage pool and compresses them:

```
rule 'COMPR1' migrate from pool 'datapool' COMPRESS('yes') where name like 'green%'
```

The following rule migrates and decompresses the same set of files:

```
rule 'COMPR1' migrate from pool 'datapool' COMPRESS('no') where name like 'green%'
```

In the following example, the first rule excludes from compression any file that ends with `.mpg` or `.jpg`. The second rule automatically compresses any file that was not accessed in the last 30 days:

```
RULE 'NEVER_COMPRESS' EXCLUDE WHERE lower(NAME) LIKE '%.mpg' OR lower(NAME) LIKE '%.jpg'
RULE 'COMPRESS_COLD' MIGRATE COMPRESS('yes') WHERE (CURRENT_TIMESTAMP - ACCESS_TIME) > (INTERVAL '30' DAYS)
```

For more information, see the following help topics:

- The topic *mmchattr command* in the *IBM Spectrum Scale: Administration and Programming Reference*

- | • *Overview of policies in the IBM Spectrum Scale: Advanced Administration Guide*
- | • *Policy rules: Syntax in the IBM Spectrum Scale: Advanced Administration Guide*
- | • *Policy rules: Syntax in the IBM Spectrum Scale: Advanced Administration Guide*

When you do file compression, you can defer the compression operation a later time. For more information, see the subtopic “Deferred file compression.”

Warnings

Doing any of the following operations while the **mmrestorefs** command is running can corrupt file data:

- Doing file compression or decompression.
- Running the **mmrestripecompress** command or the **mmrestripecompress**, either to complete a deferred file compression or decompression, or for any other reason.

Note:

- Do not run file compression or decompression while an **mmrestorefs** command is running. This warning includes compression or decompression with the **mmchattr** command or with the **mmapplypolicy** command.
- Do not run the **mmrestripecompress** or **mmrestripecompress** command while an **mmrestorefs** command is running.

Reported size of compressed files

After a file is compressed, operating system commands, such as `ls -l`, display the uncompressed size. Use `du` or the GPFs command **mmddf** to display the actual, compressed size. You can also make the **stat()** system call to find how many blocks the file occupies.

Deferred file compression

By default, the command that launches a file compression or decompression does not return until after the compression or decompression operation is completed. However, with both the **mmchattr** command and the **mmapplypolicy** command, you can defer the compression or decompression operation and have the command return as soon as it completes any other operations. By deferring compression or decompression, you can complete the operation later when the system is not heavily loaded with processes or I/O.

To defer the compression, with either command, specify the **-I defer** option. For example, the following command marks the specified file as needing compression but defers the compression operation:

```
mmchattr -I defer --compression yes trcrpt.150913.13.30.13.3518.txt
```

With the **mmapplypolicy** command, the **-I defer** option defers compression or decompression as well as data movement or deletion. For example, the following command applies the rules in the file `policyfile` but defers the file operations that are specified in the rules, including compression or decompression:

```
mmapplypolicy fs1 -P policyfile -I defer
```

To complete a deferred compression or decompression, run the **mmrestripecompress** command or the **mmrestripecompress** command with the **-z** option. (Do not run either of these commands if an **mmrestorefs** command is running. See the warnings in the preceding subtopic “Warnings.”) The following command completes the deferred compression or decompression of the specified file:

```
mmrestripecompress -z trcrpt.150913.13.30.13.3518.txt
```

Indicators of file compression or decompression

The **mmfattr** command displays two indicators that together describe the state of compression or decompression of the specified file:

COMPRESSED

The **mmlsattr** command displays the **COMPRESSED** indicator on the Misc attributes line of its output. See the example of **mmlsattr** output in Figure 1. If present, **COMPRESSED** indicates that the file is compressed or is marked for deferred compression. If absent, the absence indicates that the file is uncompressed or is marked for deferred decompression.

This indicator reflects the state of the GPFS_IWINFLAG_COMPRESSED flag in the **gpfs_iattr64_t** structure of the inode of the file. For more information about this structure, see the topic *gpfs_iattr64_t_structure* in the *IBM Spectrum Scale: Administration and Programming Reference*.

illCompressed

The **mmlsattr** command displays the **illCompressed** indicator on the flags line of its output. See Figure 1. If present, **illCompressed** indicates that the file is marked for compression or decompression but that compression or decompression is not completed. If absent, the absence indicates that compression or decompression is completed. For more information about this structure, see the topic *gpfs_iattr64_t_structure* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This indicator reflects the state of the GPFS_IAFLAG_ILLCOMPRESSED flag in the **gpfs_iattr64_t** structure of the inode of the file. For more information about this structure, see the topic *gpfs_iattr64_t_structure* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Note: Some file system events can cause the **illCompressed** flag to be set. Consider the following examples:

- When data is written into an already compressed file, the existing data remains compressed but the new data is uncompressed. The **illCompressed** flag is set for this file.
- When a compressed file is memory-mapped, the memory-mapped area of the file is decompressed before it is read into memory. The **illCompressed** flag is set for this file.

For more information, see the subtopic “Updates to compressed files” on page 39.

In the following example, the output from the **mmlsattr** command includes both the **COMPRESSED** indicator and the **illCompressed** indicator. This combination indicates that the file is marked for compression but that compression is not completed:

```
mmlsattr -L green02.51422500687
file name:          green02.51422500687
metadata replication: 1 max 2
data replication:   2 max 2
immutable:         no
appendOnly:        no
flags:             illCompressed
storage pool name: datapool
fileset name:      root
snapshot name:
creation time:     Wed Jan 28 19:05:45 2015
Misc attributes:   ARCHIVE COMPRESSED
Encrypted:         no
```

Figure 1. Compression and decompression indicators

Together the **Compressed** and **illCompressed** indicators indicate the compressed or uncompressed state of the file. See the following table:

Table 4. **COMPRESSED** and **illCompressed** indicators

State of the file	COMPRESSED is displayed?	illCompressed is displayed?
Uncompressed.	No	No
Decompression is not complete.	No	Yes
Compressed.	Yes	No

Table 4. COMPRESSED and illCompressed indicators (continued)

State of the file	COMPRESSED is displayed?	illCompressed is displayed?
Compression is not complete.	Yes	Yes

Updates to compressed files

When a compressed file is updated by a write operation, the file system automatically decompresses the region of the file that contains the affected data and sets the **illCompressed** flag. The file system then makes the update. To recompress the file, run the **mmrestripefile** command with the **-z** option, as in the following example:

```
mmrestripefile -z trcrpt.150913.13.30.13.3518.txt
```

The **mmrestorefs** command can cause a compressed file in the active file system to become decompressed if it is overwritten by the restore process. To recompress the file, run the **mmrestripefile** command with the **-z** option.

For more information, see the preceding subtopic “Deferred file compression” on page 37.

File compression and memory mapping

You can memory-map a file that is already compressed. The file system automatically decompresses the paged-in region and sets the **illCompressed** flag. To recompress the file, run the **mmrestripefile** command with the **-z** option.

As a convenience, the file system does not compress an uncompressed file or partially decompressed file if the file is memory-mapped. Compressing the file would not be not effective because memory mapping decompresses any compressed data in the regions that are paged in.

File compression and direct I/O

You can open a compressed file for Direct I/O, but internally the direct I/O reads and writes are replaced by buffered decompressed I/O reads and writes.

As a convenience, the file system does not compress a file that is opened for Direct I/O. Compressing the file would not be effective because direct I/O would be replaced by buffered decompressed I/O.

Backing up and restoring compressed files

Files are decompressed when they are moved out of storage that is directly managed by IBM Spectrum Scale. This fact affects file backups by products like IBM Spectrum Protect, Tivoli Storage Manager for Space Management (HSM), Linear Tape File System™ (LTFs), Transparent Cloud Tiering (TCT), and others. When you back up a file with these products, the file system decompresses the file data inline when it is read by the backup agent. The file system also sets the **illCompressed** flag in the file properties. The backed-up file data is not compressed.

When you restore a file to the IBM Spectrum Scale file system, the file data remains uncompressed but the **illCompressed** flag is still set. You can recompress the file by running **mmrestripefs** or **mmrestripefile** with the **-z** option.

Limitations

Notice the restrictions stated in the preceding subtopics:

- “File compression and memory mapping”
- “File compression and direct I/O”

- “Backing up and restoring compressed files” on page 39

File compression has the following limitations:

- File compression in this release is designed to be used only for compressing cold data or write-once objects and files. Compressing other types of data can result in performance degradation. File compression uses the zlib data compression library and favors saving space over speed.
- File compression processes consecutive segments of a file. For each segment, file compression calculates the potential savings in space. If the space savings are less than a certain threshold (10%), file compression does not compress the file segment but skips to the next segment.
- Direct I/O is not supported for compressed files.
- The following operations are not supported:
 - Compressing files in snapshots
 - Compressing a clone
 - Compressing files in an AFM cache site or in an AFM-based asynchronous Disaster Recovery (DR) fileset.
 - Compressing small files (files that consume fewer than two subblocks, compressing small files into an inode).
 - Compressing files other than regular files, such as directories.
 - Compressing files in a File Placement Optimizer (FPO) environment or in horizontal storage pools.
 - Cloning a compressed file
- On Windows:
 - Compression or decompression with the **mmapplypolicy** command is not supported.
 - Compression of files in Windows hyper allocation mode is not supported.
 - The following Windows APIs are not supported:
 - FSCTL_SET_COMPRESSION to enable/disable compression on a file
 - FSCTL_GET_COMPRESSION to retrieve compression status of a file
 - In Windows Explorer, in the Advanced Attributes window, the compression feature is not supported.

Setting the Quality of Service for I/O operations (QoS)

Use the QoS capability to limit the effect of I/O-intensive GPFS commands on overall system performance.

You can use the QoS capability to prevent I/O-intensive, long-running GPFS commands from dominating file system performance and significantly delaying other tasks. Commands like the examples in Figure 2 can generate hundreds or thousands of requests for I/O operations per second. The high demand can greatly slow down normal tasks that are competing for the same I/O resources.

```
mmrestripefs fsname -N  
mmapplypolicy fsname -N all ...
```

Figure 2. Examples of long-running, IO-intensive GPFS commands

The I/O intensive, potentially long-running GPFS commands are collectively called *maintenance commands* and are listed in the help topic for the *mmchqos* command in the *IBM Spectrum Scale: Administration and Programming Reference*.

With QoS configured, you can assign an instance of a maintenance command to a QoS class that has a lower I/O priority. Although the instance now takes longer to run to completion, normal tasks have greater access to I/O resources and run more quickly.

Overview of using QoS

The following steps provide an overview of how to use QoS. For more information, see the descriptions of the QoS commands:

- *mmchqos* command in the *IBM Spectrum Scale: Administration and Programming Reference*
- *mmlsqos* command in the *IBM Spectrum Scale: Administration and Programming Reference*

In this overview, assume that the file system `fs0` contains 5 nodes and has two storage pools: the system storage pool (`system`) and another storage pool `sp1`.

1. Monitor your file system with the **mmlsqos** command to determine its maximum capacity in I/O operations per second (IOPS). Follow these steps:
 - a. Enable QoS without placing any limits on I/O consumption. The following command sets the QoS classes of both storage pools to **unlimited**:

Table 5. Set QoS classes to **unlimited**

Storage pool	QoS class: maintenance	QoS class: other
system	unlimited	unlimited
sp1	unlimited	unlimited

```
mmchqos fs0 --enable --reset
```

- b. Run some maintenance commands that drive I/O on all nodes and disks.
 - c. Run the **mmlsqos** command to observe how many IOPS are consumed:


```
mmlsqos fs0 --seconds 60
```
2. Run the **mmchqos** command to allocate the available IOPS among the storage pools.
 - a. Allocate a smaller share of IOPS to the **maintenance** class, perhaps 15 percent. For example, if you determined in Step 1 that the maximum is 10,000 IOPS, then you might allocate 1500 IOPS to the maintenance class.

If there is more than one storage pool, then divide the IOPS among the maintenance classes of the storage pools. In this overview, suppose that you decide to allocate 1000 IOPS to the maintenance class of the system pool and 500 IOPS to the maintenance class of the `sp1` storage pool. See the second column of the table below.

Note: Make sure that the virtual storage Logical Unit Numbers (LUNs) of different storage pools do not map to the same physical devices.

QoS divides specific allocations of IOPS evenly among the nodes in the file system. In this overview there are 5 nodes. So QoS allocates 200 IOPS to the **maintenance** class of the system pool and 100 IOPS to the **maintenance** class of the `sp1` storage pool on each node.

- b. Allocate the remaining IOPS to the **other** classes. It is a good idea to accomplish this task by setting **other** to **unlimited** in each storage class. Then normal tasks can absorb all the IOPS of the system when no maintenance commands are running. See the third column of the following table:

Table 6. Allocate the available IOPS

Storage pool	QoS class: maintenance	QoS class: other
system	1000 IOPS (200 IOPS per node)	unlimited
sp1	500 IOPS (100 IOPS per node)	unlimited

The command is on one line:

```
mmchqos fs0 --enable pool=system,maintenance=1000IOPS,other=unlimited
pool=sp1,maintenance=500IOPS,other=unlimited
```

3. When you run a maintenance command, QoS by default assigns it to the **maintenance** class:


```
mmdeldisk fs0 nsd12
```

All maintenance command instances that are running at the same time and that access the same storage pool compete for the IOPS that you allocated to the maintenance class of that storage pool. If the IOPS limit of the class is exceeded, then QoS queues the extra I/O requests until more IOPS become available.

To run a maintenance command without I/O restrictions, you can explicitly assign it to the **other** class:

```
mmdeldisk fs0 nsd12 --qos other
```

4. You can disable QoS at any time without losing your IOPS allocations:

```
mmchqos fs0 --disable
```

When you reenables QoS it starts applying the allocations again:

```
mmchqos fs0 --enable
```

5. You can change the IOPS allocations at any time. The following command is on one line:

```
mmchqos fs0 --enable pool=system,maintenance=750IOPS,other=unlimited  
pool=sp1,maintenance=750IOPS,other=unlimited
```

When you change allocations, mount the file system, or reenables QoS, a brief delay due to reconfiguration occurs before QoS starts applying allocations.

6. To monitor the consumption of IOPS while a maintenance command is running, run the **mmlsqos** command. The following command displays the statistics for the preceding 60 seconds during which a maintenance command was running:

```
mmlsqos fs0 --seconds 60
```

See also

- | • “mmchqos command” on page 396
- | • “mmlsqos command” on page 556

Restripping a GPFS file system

Writing data into a GPFS file system correctly stripes the file. However, if you have added disks to a GPFS file system that are seldom updated, use the **mmrestripefs** command to restripe the file system to achieve maximum performance. You can also use **mmrestripefs** to perform any incomplete or deferred file compression or decompression.

Restripping offers the opportunity to specify useful options in addition to rebalancing (**-b** option). Re-replicating (**-r** or **-R** option) provides for proper replication of all data and metadata. If you use replication, this option is useful to protect against additional failures after losing a disk. For example, if you use a replication factor of 2 and one of your disks fails, only a single copy of the data would remain. If another disk then failed before the first failed disk was replaced, some data might be lost. If you expect delays in replacing the failed disk, you could protect against data loss by suspending the failed disk using the **mmchdisk** command and re-replicating. This would assure that all data existed in two copies on operational disks.

If files are assigned to one storage pool, but with data in a different pool, the placement (**-p**) option will migrate their data to the correct pool. Such files are referred to as ill-placed. Utilities, such as the **mmchattr** command or policy engine, may change a file's storage pool assignment, but not move the data. The **mmrestripefs** command may then be invoked to migrate all of the data at once, rather than migrating each file individually. Note that the rebalance (**-b**) option also performs data placement on all files, whereas the placement (**-p**) option rebalances only the files that it moves.

If you do not replicate all of your files, the migrate (**-m**) option is useful to protect against data loss when you have an advance warning that a disk may be about to fail, for example, when the error logs show an excessive number of I/O errors on a disk. Suspending the disk and issuing the **mmrestripefs** command with the **-m** option is the quickest way to migrate only the data that would be lost if the disk failed.

If you do not use replication, the **-m** and **-r** options are equivalent; their behavior differs only on replicated files. After a successful re-replicate (**-r** option) all suspended disks are empty. A migrate operation, using the **-m** option, leaves data on a suspended disk as long as at least one other replica of the data remains on a disk that is not suspended. Restripping a file system includes re-replicating it; the **-b** option performs all the operations of the **-m** and **-r** options.

Use the **-z** option to perform any deferred or incomplete compression or decompression of files in the file system.

Consider the necessity of restripping and the current demands on the system. New data which is added to the file system is correctly striped. Restripping a large file system requires extensive data copying and may affect system performance. Plan to perform this task when system demand is low.

If you are sure you want to proceed with the restripe operation:

1. Use the **mmchdisk** command to suspend any disks to which you *do not* want the file system restripped. You may want to exclude disks from file system restripping because they are failing. See “Changing GPFS disk states and parameters” on page 163.
2. Use the **mmlsdisk** command to assure that all disk devices to which you *do* want the file system restripped are in the up/normal state. See “Displaying GPFS disk states” on page 162.

Specify the target file system with the **mmrestripefs** command. For example, to rebalance (**-b** option) file system **fs1** after adding an additional RAID device, enter:

```
mmrestripefs fs1 -b
```

The system displays information similar to:

```
Scanning file system metadata, phase 1 ...
 19 % complete on Wed Mar 14 21:28:46 2012
 100 % complete on Wed Mar 14 21:28:48 2012
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scanning file system metadata for spl storage pool
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 100.00 % complete on Wed Mar 14 21:28:55 2012
Scan completed successfully.
```

Note: Rebalancing of files is an I/O intensive and time consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance your file system over time, without the cost of the rebalancing.

For complete usage information, see “**mmrestripefs** command” on page 642.

Querying file system space

Although you can use the **df** command to summarize the amount of free space on all GPFS disks, the **mmddf** command is useful for determining how well-balanced the file system is across your disks. (Also, the output from **mmddf** can be more up to date than the output from **df**.) Additionally, you can use the **mmddf** command to diagnose space problems that might result from fragmentation.

Note: The **mmddf** command may require considerable metadata I/O, and should be run when the system load is light.

Specify the file system you want to query with the **mmdf** command. For example, to query available space on all disks in the file system **fs1**, enter:

```
mmdf fs1
```

The system displays information similar to:

disk name	disk size in KB	failure holds group metadata	holds data	free KB in full blocks	free KB in fragments

Disks in storage pool: system (Maximum disk size allowed is 122 GB)					
hd16vsdn10	17793024	-1 yes	yes	17538560 (99%)	1728 (0%)
hd3vsdn01	8880128	2 yes	yes	8658176 (98%)	1600 (0%)
hd4vsdn01	8880128	2 yes	yes	8616448 (97%)	1384 (0%)
hd15vsdn10	17793024	10 yes	yes	17539584 (99%)	1664 (0%)
hd13vsdn02	8880128	4001 yes	yes	8663552 (98%)	1776 (0%)
hd8vsdn01	8880128	4002 yes	yes	8659200 (98%)	1936 (0%)
hd5vsdn01	8880128	4002 yes	yes	8654848 (97%)	1728 (0%)
hd33n09	17796008	4003 yes	yes	17540864 (99%)	2240 (0%)

(pool total)	257800488			252091136 (98%)	46928 (0%)

Disks in storage pool: fs1sp1 (Maximum disk size allowed is 122 GB)					
hd30n01	8897968	8 no	yes	8895488 (100%)	424 (0%)
hd31n01	8897968	8 no	yes	8895488 (100%)	424 (0%)

(pool total)	17795936			17790976 (100%)	848 (0%)

(data)	266716296			261222144 (98%)	44576 (0%)
(metadata)	248920360			243217408 (98%)	46048 (0%)
=====					
(total)	275596424			269882112 (98%)	47776 (0%)

Inode Information

```
-----
Number of used inodes:      9799
Number of free inodes:     4990393
Number of allocated inodes: 5000192
Maximum number of inodes:  5000192
```

For complete usage information, see “mmdf command” on page 470.

Querying and reducing file system fragmentation

Disk fragmentation within a file system is an unavoidable condition. When a file is closed after it has been written to, the last logical block of data is reduced to the actual number of subblocks required, thus creating a fragmented block.

In order to **write** to a file system, free full blocks of disk space are required. Due to fragmentation, it is entirely possible to have the situation where the file system is not full, but an insufficient number of free full blocks are available to **write** to the file system. Replication can also cause the copy of the fragment to be distributed among disks in different failure groups. The **mmdefragfs** command can be used to query the current fragmented state of the file system and reduce the fragmentation of the file system.

In order to reduce the fragmentation of a file system, the **mmdefragfs** command migrates fragments to free space in another fragmented disk block of sufficient space, thus creating a free full block. There is no requirement to have a free full block in order to run the **mmdefragfs** command. The execution time of the **mmdefragfs** command depends on the size and allocation pattern of the file system. For a file system with a large number of disks, the **mmdefragfs** command will run through several iterations of its algorithm, each iteration compressing a different set of disks. Execution time is also dependent on how fragmented the file system is. The less fragmented a file system, the shorter time for the **mmdefragfs** command to execute.

The fragmentation of a file system can be reduced on all disks which are not suspended or stopped. If a disk is suspended or stopped, the state of the disk, not the utilization information, will be displayed as output for the **mmdefragfs** command.

The **mmdefragfs** command can be run on both a mounted or an unmounted file system, but achieves best results on an unmounted file system. Running the command on a mounted file system can cause conflicting allocation information and consequent retries to find a new free subblock of the correct size to store the fragment in.

Querying file system fragmentation

To query the current status of the amount of fragmentation for a file system, specify the file system name along with the **-i** option on the **mmdefragfs** command.

For example, to display the current fragmentation information for file system **fs0**, enter:

```
mmdefragfs fs0 -i
```

The system displays information similar to:

```
"fs0"    10304 inodes:    457 allocated / 9847 free
```

disk name	disk size in nSubblk	free subblk in full blocks	free subblk in fragments	% free blk	% blk util
gpfs68nsd	4390912	4270112	551	97.249	99.544
gpfs69nsd	4390912	4271360	490	97.277	99.590
(total)	8781824	8541472	1041		99.567

For complete usage information, see “mmdefragfs command” on page 443.

Reducing file system fragmentation

You can reduce the amount of fragmentation for a file system by issuing the **mmdefragfs** command, with or without a desired block usage goal.

For example, to reduce the amount of fragmentation for file system **fs1** with a goal of 100% utilization, enter:

```
mmdefragfs fs1 -u 100
```

The system displays information similar to:

```
Defragmenting file system 'fs1'...
```

```
Defragmenting until full block utilization is 98.00%, currently 97.07%
27.35 % complete on Tue May 26 14:25:42 2009 ( 617882 inodes 4749 MB)
82.65 % complete on Tue May 26 14:26:02 2009 ( 1867101 inodes 10499 MB)
89.56 % complete on Tue May 26 14:26:23 2009 ( 2023206 inodes 14296 MB)
90.01 % complete on Tue May 26 14:26:43 2009 ( 2033337 inodes 17309 MB)
90.28 % complete on Tue May 26 14:27:03 2009 ( 2039551 inodes 19779 MB)
91.17 % complete on Tue May 26 14:27:23 2009 ( 2059629 inodes 23480 MB)
91.67 % complete on Tue May 26 14:27:43 2009 ( 2070865 inodes 26760 MB)
92.51 % complete on Tue May 26 14:28:03 2009 ( 2089804 inodes 29769 MB)
93.12 % complete on Tue May 26 14:28:23 2009 ( 2103697 inodes 32649 MB)
93.39 % complete on Tue May 26 14:28:43 2009 ( 2109629 inodes 34934 MB)
95.47 % complete on Tue May 26 14:29:04 2009 ( 2156805 inodes 36576 MB)
95.66 % complete on Tue May 26 14:29:24 2009 ( 2160915 inodes 38705 MB)
95.84 % complete on Tue May 26 14:29:44 2009 ( 2165146 inodes 40248 MB)
96.58 % complete on Tue May 26 14:30:04 2009 ( 2181719 inodes 41733 MB)
96.77 % complete on Tue May 26 14:30:24 2009 ( 2186053 inodes 43022 MB)
96.99 % complete on Tue May 26 14:30:44 2009 ( 2190955 inodes 43051 MB)
97.20 % complete on Tue May 26 14:31:04 2009 ( 2195726 inodes 43077 MB)
```

```

97.40 % complete on Tue May 26 14:31:24 2009 ( 2200378 inodes 43109 MB)
97.62 % complete on Tue May 26 14:31:44 2009 ( 2205201 inodes 43295 MB)
97.83 % complete on Tue May 26 14:32:05 2009 ( 2210003 inodes 43329 MB)
97.85 % complete on Tue May 26 14:32:25 2009 ( 2214741 inodes 43528 MB)
97.86 % complete on Tue May 26 14:32:55 2009 ( 2221888 inodes 43798 MB)
97.87 % complete on Tue May 26 14:33:35 2009 ( 2231453 inodes 44264 MB)
97.88 % complete on Tue May 26 14:34:26 2009 ( 2243181 inodes 45288 MB)
100.00 % complete on Tue May 26 14:35:10 2009

```

disk name	free subblk in full			free subblk in fragments		% free blk		% blk util	
	before	after	blk freed	before	after	before	after	before	after
nsd32	277504	287840	323	12931	2183	84.69	87.84	96.05	99.33
nsd33	315232	315456	7	580	185	96.20	96.27	99.82	99.94
nsd21	301824	303616	56	2481	666	92.11	92.66	99.24	99.80
nsd34	275904	285920	313	13598	3159	84.20	87.26	95.85	99.04
nsd30	275840	285856	313	13348	2923	84.18	87.24	95.93	99.11
nsd19	278592	288832	320	12273	1874	85.02	88.14	96.25	99.43
nsd31	276224	284608	262	12012	3146	84.30	86.86	96.33	99.04
(total)	2001120	2052128	1594	67223	14136			97.07	99.38

Defragmentation complete, full block utilization is 99.04%.

See the “`mmdefragfs` command” on page 443 for complete usage information.

Protecting data in a file system using backup

GPFS provides a way to back up the file system user data and the overall file system configuration information.

You can use the `mmbackup` command to back up the files of a GPFS file system or the files of an independent fileset to a Tivoli Storage Manager (TSM) server.

Alternatively, you can utilize the GPFS policy engine (`mmapplypolicy` command) to generate lists of files to be backed up and provide them as input to some other external storage manager.

The file system configuration information can be backed up using the `mmbackupconfig` command.

Note: Windows nodes do not support the `mmbackup`, `mmapplypolicy`, and `mmbackupconfig` commands.

Protecting data in a file system using the `mmbackup` command

The `mmbackup` command can be used to back up some or all of the files of a GPFS file system to Tivoli Storage Manager (TSM) servers using the TSM Backup-Archive client. After files have been backed up, you can restore them using the interfaces provided by TSM.

The `mmbackup` command utilizes all the scalable, parallel processing capabilities of the `mmapplypolicy` command to scan the file system, evaluate the metadata of all the objects in the file system, and determine which files need to be sent to backup in TSM, as well which deleted files should be expired from TSM. Both backup and expiration take place when running `mmbackup` in the incremental backup mode.

The `mmbackup` command can interoperate with regular TSM commands for backup and expire operations. However if after using `mmbackup`, any TSM incremental or selective backup or expire commands are used, `mmbackup` needs to be informed of these activities. Use either the `-q` option or the

--rebuild option in the next **mmbackup** command invocation to enable **mmbackup** to rebuild its shadow databases. (See **mmbackup Examples** in *IBM Spectrum Scale: Administration and Programming Reference*.)

These databases *shadow* the inventory of objects in TSM so that only new changes will be backed up in the next incremental **mmbackup**. Failing to do so will needlessly back up some files additional times. The shadow database can also become out of date if **mmbackup** fails due to certain TSM server problems that prevent **mmbackup** from properly updating its shadow database after a backup. In these cases it is also required to issue the next **mmbackup** command with either the **-q** option or the **--rebuild** options.

The **mmbackup** command provides:

- A full backup of all files in the specified scope.
- An incremental backup of only those files that have changed or been deleted since the last backup. Files that have changed since the last backup are updated and files that have been deleted since the last backup are expired from the TSM server.
- Utilization of a fast scan technology for improved performance.
- The ability to perform the backup operation on a number of nodes in parallel.
- Multiple tuning parameters to allow more control over each backup.
- The ability to backup the read/write version of the file system or specific global snapshots.
- Storage of the files in the backup server under their GPFS root directory path independent of whether backing up from a global snapshot or the live file system.
- Handling of unlinked filesets to avoid inadvertent expiration of files.

Note: Avoid unlinking a fileset while running **mmbackup**. If a fileset is unlinked before **mmbackup** starts, it is handled; however, unlinking a fileset during the job could result in a failure to back up changed files as well as expiration of already backed up files from the unlinked fileset.

The **mmbackup** command supports backing up GPFS file system data to multiple Tivoli Storage Manager servers. The ability to partition file backups across multiple TSM servers is particularly useful for installations that have a large number of files. For information on setting up multiple TSM servers, see “Tivoli Storage Manager requirements” on page 48.

Unless otherwise specified, the **mmbackup** command backs up the current active version of the GPFS file system. If you want to create a backup of files at a specific point in time, first use the **mmcrsnapshot** command to create either a global snapshot or a fileset-level snapshot, and then specify that snapshot name for the **mmbackup -S** option. A global snapshot can be specified for either **--scope filesystem** or **--scope inodespace**. A fileset-level snapshot can only be specified with **--scope inodespace**.

If an unlinked fileset is detected, the **mmbackup** processing will issue an error message and exit. You can force the backup operation to proceed by specifying the **mmbackup -f** option. In this case, files that belong to unlinked filesets will not be backed up, but will be removed from the expire list.

If you have file systems that were backed up using the GPFS 3.2 or earlier version of the **mmbackup** command, you will not be able to take advantage of some of the new **mmbackup** features until a new full backup is performed. See “File systems backed up using GPFS 3.2 or earlier versions of **mmbackup**” on page 49.

Protecting data in a fileset using the **mmbackup** command

The **mmbackup** command can be used to back up an independent fileset to Tivoli Storage Manager (TSM) servers using the TSM Backup-Archive client. After a fileset has been backed up, you can restore files using the interfaces provided by TSM.

When backing up an independent fileset, the **mmbackup** command backs up the current active version of the fileset. The path to the independent fileset root is specified with the *Directory* parameter of the **mmbackup** command.

If you want to create a backup of a fileset at a specific point in time, first use the **mmcrsnapshot** command to create a fileset-level snapshot. Next, specify that snapshot name for the **mmbackup -S** option along with the **--scope inodespace** option.

Tivoli Storage Manager requirements

The **mmbackup** command requires a Tivoli Storage Manager client and server environment to perform a backup operation.

For details on the supported versions of TSM, client and server installation and setup, and include and exclude lists, see the IBM Tivoli Storage Manager V6.3 documentation (www.ibm.com/support/knowledgecenter/SSGSG7_6.3.0/com.ibm.itism.ic.doc/welcome.html).

1. Ensure that the supported versions of the TSM client and server are installed. See the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
2. Ensure that the TSM server and clients are configured properly for backup operations.
3. If you are using multiple TSM servers to protect data, ensure that the TSM servers are set up properly.
4. Ensure the required **dsm.sys** and **dsm.opt** configuration files are present in the TSM configuration directory on each node used to run **mmbackup** or named in a node specification with **-N**.
5. If you want to include or exclude specific files or directories by using include-exclude lists, ensure that the lists are set up correctly before you invoke the **mmbackup** command.

The **mmbackup** command uses a TSM include-exclude list for including and excluding specific files or directories. See the Tivoli documentation for information about defining an include-exclude list.

Note: TSM interprets its include and exclude statements in a unique manner that is not precisely matched by the GPFS **mmapplypolicy** file selection language. The essential meaning of each supported include or exclude statement is followed, but the commonly used TSM idiom of excluding everything as the last statement and including selective directory or file name patterns in prior statements should not be used with GPFS and **mmbackup**. The exclusion pattern of **"/**"** is interpreted by **mmapplypolicy** to exclude everything, and no data is backed up.

A very large include-exclude list can decrease backup performance. Use wildcards and eliminate unnecessary include statements to keep the list as short as possible.

6. If more than one node will be used to perform the backup operation (**mmbackup -N** option):
 - The **mmbackup** command will verify that the TSM Backup-Archive client versions and configuration are correct before executing the backup. Any nodes that are not configured correctly will be removed from the backup operation. Ensure that TSM clients are installed and at the same version on all nodes that will invoke the **mmbackup** command or participate in parallel backup operations.
 - Ensure that TSM is aware that the various TSM clients are all working on the same file system, not different file systems having the same name on different client machines. This is accomplished by using proxy nodes for multiple nodes in the cluster. See the TSM documentation for recommended settings for GPFS cluster nodes setup.
7. Restoration of backed-up data must be done using TSM interfaces. This can be done with the client command-line interface or the TSM web client. The TSM web client interface must be made operational if you wish to use this interface for restoring data to the file system from the TSM server.
8. When more than one TSM server is referenced in the **dsm.sys** file, **mmbackup** uses all listed TSM servers by default. To use only a select TSM server or the servers that are listed in **dsm.sys**, use the **mmbackup --tsm-servers** option. When more than one TSM server is used for backup, the list and the order specified should remain constant. If additional TSM servers are added to the backup later, add them to the end of the list that is specified with the **mmbackup --tsm-servers** option.
9. TSM does not support special characters in the path names and in some cases cannot back up a path name that has special characters. A limited number of special characters are supported on TSM client

6.4.0.0 and later versions with client options **WILDCARDSARELITERAL** and **QUOTESARELITERAL**. Use these TSM options with the **mmbackup --noquote** option if you have path names with special characters. The **mmbackup** command does not back up path names containing any newline, **Ctrl+x**, or **Ctrl+y** characters. If the **mmbackup** command finds unsupported characters in the path name, it writes that path to a file called `mmbackup.unsupported.tsmserver` at the root of the **mmbackup** record directory (by default it is the root of the file system).

Attention: If you are using the TSM Backup-Archive client command line or web interface to do back up, use caution when you unlink filesets that contain data backed up by TSM. TSM tracks files by path name and does not track filesets. As a result, when you unlink a fileset, it appears to TSM that you deleted the contents of the fileset. Therefore, the TSM Backup-Archive client inactivates the data on the TSM server, which may result in the loss of backup data during the expiration process.

File systems backed up using GPFS 3.2 or earlier versions of mmbackup

GPFS 3.2 and earlier versions of the **mmbackup** command automatically created a temporary snapshot named **.mmbuSnapshot** of the specified file system, and backed up this snapshot to the TSM server. Accordingly, the files backed up by the command were stored in TSM using the `/Device/.snapshots/.mmbuSnapshot` directory path in the remote data store.

The GPFS 3.3 through GPFS 3.5.0.11 versions of the **mmbackup** command will preserve this type of processing for incremental backups until a new full backup is performed. Once a full backup is performed, **mmbackup** will store the files in TSM under their usual GPFS root directory path name; all files under `/Device/.snapshots/.mmbuSnapshot` will be marked for expiration. Until the transition to using the usual GPFS root directory path name in TSM is complete, no backups can be taken from a snapshot, other than the **mmbackup** temporary snapshot called **.mmbuSnapshot**.

Attention: Starting with GPFS 4.1, the **mmbackup** command will no longer support the `/Device/.snapshots/.mmbuSnapshot` path name format for incremental backups. After migrating to GPFS 4.1, if the older **.mmbuSnapshot** path name format is still in use, a full backup is required if a full backup has never been performed with GPFS 3.3 or later. After the full backup is performed, files will now always be stored in TSM under their usual GPFS root directory path name. All files in TSM under `/Device/.snapshots/.mmbuSnapshot` will be marked for expiration automatically after a successful backup.

The transition to using the usual GPFS root directory path name format, instead of the `/Device/.snapshots/.mmbuSnapshot` path name format permits **mmbackup** to perform a backup using any user-specified snapshot, or the live file system interchangeably.

Certain features, such as backing up from an arbitrary snapshot, cannot be used until a full backup is performed with the GPFS 3.3 or later version of the **mmbackup** command.

Migrating to mmbackup from TSM-interface-based backup

File systems that are backed up using the TSM interface can be converted to use the **mmbackup** command to take advantage of the performance offered by **mmbackup** fast scan technology.

A full backup is not required or necessary when moving from backup using the TSM interface to the **mmbackup** command.

The **mmbackup** command uses one or more shadow database files to determine changes in the file system. To convert from the TSM interface backup to **mmbackup**, one must create the shadow database file or files by using the **--rebuild** option of **mmbackup**. The rebuild option queries the existing TSM server or servers and creates a shadow database of the files currently backed up in TSM. After the shadow database file or files are generated, **mmbackup** can be used for all future incremental or full backups.

Note: If using multiple TSM servers to back up a file system, use the **mmbackup --tsm-servers** option to ensure that the proper servers participate in the backup job.

Tuning backups with the **mmbackup** command

You can tune backups with the **mmbackup** command.

The **mmbackup** command performs all its work in three major steps, and all of these steps potentially use multiple nodes and threads:

1. The file system is scanned with **mmapplypolicy**, and a list is created of every file that qualifies and should be in backup for each TSM server in use. The existing shadow database and the list generated are then compared and the differences between them yield:
 - Objects deleted recently that should be marked inactive (expire)
 - Objects modified or newly created to back up (selective)
 - Objects modified without data changes; owner, group, mode, and migration state changes to update (incremental)
2. Using the lists created in step 1, **mmapplypolicy** is run for files that should be marked inactive (expire).
3. Using the lists created in step 1, **mmapplypolicy** is run for selective or incremental backup.

The **mmbackup** command has several parameters that can be used to tune backup jobs. During the scanning phase, the resources **mmbackup** will utilize on each node specified with the **-N** parameter can be controlled:

- The **-a *IsanThreads*** parameter allows specification of the number of threads and sort pipelines each node will run during the parallel inode scan and policy evaluation. This parameter affects the execution of the high-performance protocol that is used when both the **-g** and **-N** parameters are specified. The default value is 2. Using a moderately larger number can significantly improve performance, but might strain the resources of the node. In some environments a large value for this parameter can lead to a command failure.

Tip: Set this parameter to the number of CPU *cores* implemented on a typical node in your GPFS cluster.

- The **-n *DirThreadLevel*** parameter allows specification of the number of threads that will be created and dispatched within each **mmapplypolicy** process during the directory scan phase.

During the execution phase for expire, **mmbackup** processing can be adjusted as follows:

- Automatic computation of the ideal expire bunch count. The number of objects named in each file list can be determined, separately from the number in a backup list, and automatically computed, if not specified by the user.
- As an alternative to the automatic computation, the user can control expire processing as follows:
 - The **--max-expire-count** parameter can be used to specify a bunch-count limit for each **dsmc expire** command. This parameter cannot be used in conjunction with **-B**.
 - The **--expire-threads** parameter can be used to control how many threads run on each node running **dsmc expire**. This parameter cannot be used in conjunction with **-m**.

During the execution phase for backup, **mmbackup** processing can be adjusted as follows:

- Automatic computation of ideal backup bunch count. The number of objects named in each file list can be determined, separately from the number in an expire list, and automatically computed, if not specified by the user.
- As an alternative to the automatic computation, the user can control backup processing as follows:
 - The **--max-backup-count** parameter can be used to specify a bunch-count limit for each **dsmc selective** or **dsmc incremental** command. This parameter cannot be used in conjunction with **-B**.

- The **--backup-threads** parameter can be used to control how many threads run on each node running backup. This parameter cannot be used in conjunction with **-m**.
- The **--max-backup-size** parameter can be used to further limit the size of a backup bunch by the overall size of all files listed in any single bunch list.

For more information on the **mmbackup** tuning parameters, see “mmbackup command” on page 281.

MMBACKUP_PROGRESS_CALLOUT environment variable

The **MMBACKUP_PROGRESS_CALLOUT** environment variable specifies the path to a program or script to be called during **mmbackup** execution with a formatted argument.

The **\$progressCallOut** function is executed if the path **\$progressCallOut** names a valid, executable file and one of the following is true:

- The message class provided with this message is 0.
Or
- At least **\$progressInterval** seconds has elapsed.
Or
- The **\$progressContent** mask has a bit set which matches a bit set in the message class provided with this message.

The **\$progressCallOut** function is executed during **mmbackup** with a single argument consisting of the following colon-separated values:

```
"$JOB:$FS:$SERVER:$NODENAME:$PHASE:$BCKFILES:$CHGFILES:$EXPFILES:\
$FILESBACKEDUP:$FILESEXPIRED:$ERRORS:$TIME"
```

Where:

JOB

Specifies the literal backup string to identify this component.

FS Specifies the file system device name.

SERVER

Specifies the TSM server currently used for backup.

NODENAME

Specifies the name of the node where **mmbackup** was started.

PHASE

Specifies either **synchronizing**, **scanning**, **selecting files**, **expiring**, **backing up**, **analyzing**, or **finishing**.

BCKFILES

Specifies the total number of files already backed up, or stored, on the TSM server. Starts as the count of all normal mode records in all the current shadow databases in use. If **QUERY** is being executed, it will start as the count of files found on the TSM server. It will stay constant until the backup job is complete.

CHGFILES

Specifies the number of changed files. This value starts as 0 and changes to the total number of changed files destined for the current server, and then stays at that value.

EXPFILES

Specifies the number of expired files. This value starts as 0 and changes to the total number of files marked for expiration at the current server, and then stays at that value.

FILESBACKEDUP

Specifies the number of files that were backed up during this backup job. This value remains 0 until phase **backing up** is reached, and then it increases until **dsmsc** finishes. This value increases while

dsmc selective jobs are running, and is calculated by TSM output. If the backup job fails before completion, some output may indicate files backed up but not counted. This value always increases.

FILEEXPIRED

Specifies the number of files that expired during this **expire** job. This value remains 0 until phase **expiring** is reached, and then it increases until **dsmc** finishes. This value increases while **dsmc** expire jobs are running, and is calculated by TSM output. If the backup job fails before completion, some output may indicate files expired but not counted. This value always increases.

ERRORS

Specifies the number of errors, not warnings or informational messages, that occurred during processing.

TIME

Specifies the time stamp as a **ctime** or number of seconds since the Epoch.

Backing up a file system using the GPFS policy engine

If Tivoli Storage Manager is not available, you can use the fast scan capabilities of the GPFS policy engine to generate lists of files to be backed up and provide them as input to some other external storage manager.

This process typically includes:

- Creating a policy file with LIST rules and associated criteria to generate the desired lists
- Optionally, creating a snapshot to obtain a consistent copy of the file system at a given point in time
- Running the **mmapplypolicy** command to generate the lists of files to back up
- Invoking the external storage manager to perform the actual backup operation

For more information on GPFS policies and rules refer to *Information Lifecycle Management for GPFS* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Backing up file system configuration information

The **mmbackupconfig** command can be used to back up vital file system configuration information. This information can later be used to restore the layout and major characteristics of the file system.

The **mmbackupconfig** command creates a file that includes:

- Disk information (NSD names, sizes, failure groups)
- Storage pool layout
- Filesets and junction points
- Policy file rules
- Quota settings and current limits
- File system parameters (block size, replication factors, number of inodes, default mount point, and so on)

The output file generated by the **mmbackupconfig** command is used as input to the **mmrestoreconfig** command.

Note: The **mmbackupconfig** command only backs up the file system configuration information. It does not back up any user data or individual file attributes.

It is recommended that you store the output file generated by **mmbackupconfig** in a safe location.

Using APIs to develop backup applications

You can develop backup applications using APIs

IBM has supplied a set of subroutines that are useful to create backups or collect information about all files in a file system. Each subroutine is described in *Programming interfaces in IBM Spectrum Scale: Administration and Programming Reference*. These subroutines are more efficient for traversing a file system, and provide more features than the standard POSIX interfaces. These subroutines operate on a global snapshot or on the active file system. They have the ability to return all files, or only files that have changed since some earlier snapshot, which is useful for incremental backup.

A typical use of these subroutines is the following scenario:

1. Create a global snapshot using the **mmcrsnapshot** command. For more information on snapshots, see the *IBM Spectrum Scale: Advanced Administration Guide*.
2. Open an inode scan on the global snapshot using the **gpfs_open_inodescan()** or **gpfs_open_inodescan64()** subroutine.
3. Retrieve inodes using the **gpfs_next_inode()** or **gpfs_next_inode64()** subroutine.
4. Read the file data:
 - a. Open the file using the **gpfs_iopen()** or **gpfs_iopen64()** subroutine.
 - b. Read the file using the **gpfs_iread()**, **gpfs_ireadx()**, **gpfs_ireaddir()**, or **gpfs_ireaddir64()** subroutines.
 - c. Close the file using the **gpfs_iclose()** subroutine.

The **gpfs_ireadx()** subroutine is more efficient than **read()** or **gpfs_iread()** for sparse files and for incremental backups. The **gpfs_ireaddir()** or **gpfs_ireaddir64()** subroutine is more efficient than **readdir()**, because it returns file type information. There are also subroutines for reading symbolic links, **gpfs_ireadlink()** or **gpfs_ireadlink64()** and for accessing file attributes, **gpfs_igetattrs()**.

Scale Out Backup and Restore (SOBAR)

Scale Out Backup and Restore (SOBAR) is a specialized mechanism for data protection against disaster only for GPFS file systems that are managed by Tivoli Storage Manager (TSM) Hierarchical Storage Management (HSM). For such systems, the opportunity exists to *premigrate* all file data into the HSM storage and take a snapshot of the file system structural metadata, and save a backup image of the file system structure. This metadata image backup, consisting of several image files, can be safely stored in the backup pool of the TSM server and later used to restore the file system in the event of a disaster.

The SOBAR utilities include the commands **mmbackupconfig**, **mmrestoreconfig**, **mmimgbackup**, and **mmimgrestore**. The **mmbackupconfig** command will record all the configuration information about the file system to be protected and the **mmimgbackup** command performs a backup of GPFS file system metadata. The resulting configuration data file and the metadata image files can then be copied to the TSM server for protection. In the event of a disaster, the file system can be recovered by recreating the necessary NSD disks, restoring the file system configuration with the **mmrestoreconfig** command, and then restoring the image of the file system with the **mmimgrestore** command. The **mmrestoreconfig** command must be run prior to running the **mmimgrestore** command. SOBAR will reduce the time needed for a complete restore by utilizing all available bandwidth and all available nodes in the GPFS cluster to process the image data in a highly parallel fashion. It will also permit users to access the file system before all file data has been restored, thereby minimizing the file system down time. Recall from HSM of needed file data is performed automatically when a file is first accessed.

These commands cannot be run from a Windows node.

For the full details of the SOBAR procedures and requirements, see *Scale Out Backup and Restore (SOBAR)* in *IBM Spectrum Scale: Administration and Programming Reference*.

Scheduling backups using Tivoli Storage Manager scheduler

The Tivoli Storage Manager scheduler typically utilizes the TSM Backup-Archive client backup commands that should be avoided in the IBM Spectrum Scale setup. Instead, you can configure a Tivoli Storage Manager client schedule to call a script as described in the following steps.

For scheduled events to occur on the client, you must configure the client scheduler to communicate with the Tivoli Storage Manager server. This is in addition to the following steps. For example, you might need to start the `dsmcad` service or add `MANAGEDSERVICES` schedule to the corresponding Tivoli Storage Manager stanza in `dsm.sys` on the client node. For more information, see *Configuring the scheduler* in the Tivoli Storage Manager documentation on IBM Knowledge Center.

For the following steps, these example values are assumed:

```
client-node-proxyname (asnodename)           => proxy-cluster1
Node to be used for the schedule (aka nodename) => gpfs-nod1
tsm server name                               => tsm1
file system to be backed up                   => gpfs0
global snapshot name (created for backup job) => BKUPsnap
schedule name on the TSM server               => proxy-cluster1_sched
```

1. On the Tivoli Storage Manager server, define the schedule using the following command.

```
define schedule standard proxy-cluster1_sched type=client action=command objects=/usr/bin/my-mmbackup-script.sh starttime=05:00:00 startdate=today
```
2. On the Tivoli Storage Manager server, associate the schedule with the IBM Spectrum Scale proxy node using the following command.

```
define association standard proxy-cluster1_sched proxy-cluster1
```
3. Create the backup script on the IBM Spectrum Scale node.

Note: The following example script must be extended to log the output into files so that verification or troubleshooting can be done afterwards. Additional options such as `--noquote` might be needed depending on the specific needs of the environment.

```
#!/bin/bash
/usr/lpp/mmfs/bin/mmcrsnapshot gpfs0 BKUPsnap
/usr/lpp/mmfs/bin/mmbackup gpfs0 -t incremental --tsm-servers tsm1
/usr/lpp/mmfs/bin/mmdelsnapshot gpfs0 BKUPsnap
```

4. On one of the Tivoli Storage Manager client nodes, verify the schedule using the following command.

```
dsmc q sched
```

Configuration reference for using Tivoli Storage Manager with IBM Spectrum Scale

When using the Tivoli Storage Manager client in an IBM Spectrum Scale environment, several options in the `dsm.sys` and `dsm.opt` configuration files need to be taken into consideration.

Note: Refer to the latest Tivoli Storage Manager documentation on IBM Knowledge Center for the latest information on the mentioned settings.

Options in the Tivoli Storage Manager configuration file `dsm.sys`

This topic describes the options in the Tivoli Storage Manager configuration file `dsm.sys`.

Important: While the Tivoli Storage Manager client configuration file `dsm.sys` can contain node specific information, it cannot simply be copied from node to node without touching or correcting the corresponding node specific information.

Exclude or include options

File path name patterns that do not need to be backed up might be excluded by corresponding exclude statements. For example, temporary files. While Tivoli Storage Manager provides options for excluding and including, the usage of include options must be avoided when **mmbackup** is used. The reason is that **mmbackup** processing works properly with exclude statements but misinterpretations can arise when both, include and exclude, options are used together and in worst case have overlapping pattern sequences.

Note: Defining a large number of exclude rules can negatively impact the performance of backup.

Do not add exclude statements for snapshots as snapshots are specially handled automatically by **mmbackup** and Tivoli Storage Manager options when needed.

mmbackup excludes the following folders from the scan by default and these need not be explicitly excluded in the `dsm.sys` file or on the Tivoli Storage Manager server:

- `.mmbackup*` - folder in location specified by **MMBACKUP_RECORD_ROOT** such as `/ibm/gpfs0/.mmbackupCfg`
- `.mmLockDir` - folder in the root of the file system
- `.SpaceMan` - folder anywhere in the file system
- `.TsmCacheDir` - folder anywhere in the file system

Special consideration is needed when Tivoli Storage Manager server management class definitions are used. The corresponding include statements must be applied to any `dsm.sys` and not applied on the Tivoli Storage Manager server.

Tivoli Storage Manager users might be familiar with dynamic management class assignments available when using Tivoli Storage Manager **dsmc** commands to backup files. This is not the case with **mmbackup**. Only objects identified by **mmbackup** as requiring a backup will get the needed management class update that results when the administrator alters the management class assignment in the `dsm.sys` file. Therefore, only by running a complete backup of all affected objects can a management class update be guaranteed.

Despite the recommendation to never utilize the include statements in `dsm.sys`, when a Tivoli Storage Manager management class designation is needed, the use of an include statement with the management class specification is required. In these cases, do the following steps:

1. In the Tivoli Storage Manager client configuration file `dsm.sys`, arrange the include and exclude statements as follows:
 - a. Place all the include statement first in the file along with the management class definitions.
 - b. Add the exclude statements below the include statements.
 - c. Ignore the ordering precedence rules defined in the Tivoli Storage Manager documentation regarding the ordering of these statements. Management class include statements must be listed above the exclude statements to work properly with **mmbackup**.

Note: Do not add include statements after exclude statements. Do not add exclude statements before include statements.

2. Before starting the **mmbackup** job, set the following environment variable:

```
export MMBACKUP_IGNORE_INCLUDE=1
```

Note:

- The include statements have no effect on the file system scan candidate selection in **mmapplypolicy** because the rules for include do not result in SQL statements being generated with **MMBACKUP_IGNORE_INCLUDE** activated.

- The include statements do not overrule the exclude statements which can be the case sometimes with **mmapplypolicy** policy rules generated from include and exclude formulation in Tivoli Storage Manager. It is recommended to never have overlapping patterns of any type with both include and exclude statements.

Usage of a Tivoli Storage Manager proxy node (asnodename option)

In a cluster, an operation that needs to scale is usually executed on more than one node, for example backup activities. To utilize the services of a Tivoli Storage Manager server from any of the configured cluster backup nodes, the administrator needs to specify a proxy node. This proxy node needs to be created on the Tivoli Storage Manager server similar to all other cluster backup nodes that need to be registered on the Tivoli Storage Manager server before they can be used. On all cluster backup nodes, set the **asnodename** option for the desired proxy-client node to be used in the corresponding stanza of the `dsm.sys` configuration file.

Important Tivoli Storage Manager client configuration option

Option name	Remarks	Context
ASNODENAME \$client-node-proxyname	Use the proxy node name (asnodename) instead of the cluster node name (nodename) to process cluster operations independent of a node name that is required for restore processing.	General

Options in the Tivoli Storage Manager configuration file `dsm.opt`

This topic describes the options in the Tivoli Storage Manager configuration file `dsm.opt`.

Special character handling

For IBM Spectrum Scale file systems with special characters frequently used in the names of files or directories, backup failures might occur. Known special characters that require special handling include: *, ?, ", ', carriage return, and the new line character.

In such cases, enable the Tivoli Storage Manager client options `WILDCARDSARELITERAL` and `QUOTESARELITERAL` on all nodes that are used in backup activities and make sure that the **mmbackup** option `--noquote` is used when invoking **mmbackup**.

Note: The characters control-X and control-Y are not supported by Tivoli Storage Manager. Therefore, the use of these characters in file names in IBM Spectrum Scale file systems results in these files not getting backed up to Tivoli Storage Manager.

Important Tivoli Storage Manager client configuration options

Option name	Remarks	Context
QUOTESARELITERAL [YES NO]	Requires the use of mmbackup with option <code>--noquote</code> if this is set to YES.	General
WILDCARDSARELITERAL [YES NO]	To handle the wildcard characters * and ? in file and folder names.	General

Option name	Remarks	Context
HSMDISABLEAUTOMIGDAEMONS [YES NO]	To prevent the Tivoli Storage Manager for Space Management automigration daemons from starting. Instead, the mmapplypolicy scan engine is used to identify migration candidates.	Tivoli Storage Manager for Space Management
SKIPACLUPDATECHECK [YES NO]	Requires UPDATECTIME to be enabled if this is set to YES. Using the SKIPACLUPDATECHECK option also omits checking for changes in the extended attributes (EAs) on Linux and AIX systems. Using this setting ensures that a file only gets backed up when the content of the file changes, not when only the ACL or EAs change. The backup of file after content changes then also includes the current ACL or EAs of the file.	General
SKIPACL [YES NO]	Requires UPDATECTIME to be enabled if this is set to YES. Using the skipacl option also omits EAs on Linux and AIX systems. Using this option can be considered when static ACL structures are used that can be reestablished through another tool or operation external to the Tivoli Storage Manager restore operation. If you are using this approach, ensure that the ACL is restored or established by inheritance, to avoid an unauthorized access to a recently restored file or directory. After enabling this option the ACL or EA is no longer backed up.	General
UPDATECTIME [YES NO]	This is to check the change time (ctime) attribute during a backup or archive operation. It is required to perform operations such as determining ACL changes.	General

Base Tivoli Storage Manager client configuration files for IBM Spectrum Scale usage

This topic lists all the Base Tivoli Storage Manager client configuration files and their examples for IBM Spectrum Scale.

Important: While the Tivoli Storage Manager client configuration file `dsm.sys` can contain node specific information, it cannot simply be copied from node to node without touching or correcting the corresponding node specific information.

The following are example contents of Tivoli Storage Manager configuration files.

Contents of dsm.sys

Note: Substitute the variables starting with '\$' with your own required value. See the following example values of variables.

```
SERvername $servername
  COMMMethod      TCPip
  TCPPort         $serverport
  TCPServeraddress $serverip
*  TCPAdminport   $serveradminport
  TCPBuffsize     512
  PASSWORDACCESS  generate
*  Place your exclude rules here or configure as cloptset on TSM server
  ERRORLOGName    $errorlog
  ASNODENAME      $client-node-proxyname
  NODENAME        $localnodename
```

Example values of variables used in dsm.sys

```
serverport=1500
serverip=myTSMserver.mydomain.org      OR   serverip=1.2.3.4
serveradminport=1526
errorlog=/var/log/mylogs/dsmerror.log
client-node-proxyname=proxy-cluster1
localnodename=gpfs-node1
```

Contents of dsm.opt

```
* Special character test flags
QUOTESARELITERAL YES
WILDCARDSARELITERAL YES
* to take traces just remove the * from the next two lines:
*TRACEFLAG SERVICE
*TRACEFILE /tmp/tsmtrace.txt
```

Contents of dsm.opt when Tivoli Storage Manager for Space Management is used

```
* HSM: Write extObjID to DMAPi attribute 'IBMexID' for migrated/pre-migrated files
HSMEXTOBJIDATTR yes
* HSM: Deactivate HSM Automigration and Scout search engine as this will be done by GPFS
HSMDISABLEAUTOMIGDAEMONS YES
* HSM file aggregation of small files
HSMGROUPedmigrate yes
* HSM: Determines if files that are less than 2 minutes old can be migrated during selective migration
hsmenableimmediatemigrate yes
```

Restoring a subset of files or directories from a local file system snapshot

You can restore a subset of files or directories from a local snapshot of a file system in case of accidental deletion.

Ensure the following before you begin:

- You have the full path to the files or directories that you want to restore. The path must include the file system to which these files or directories belong.
- You know which snapshot contains the files or directories that you want to restore.
- You have created a restore directory to which these files or directories are to be restored to avoid accidentally overwriting files or directories.

Use these steps to restore files or directories from a local file system snapshot.

1. Use the `mmlsnapshot device` command to list the snapshots in the file system and make a note of the snapshot that contains the files and directories that you want to restore.

device is the name of the file system.

```
# mmlsnapshot fs1
```

Snapshots in file system fs1:

Directory	SnapId	Status	Created	Fileset
fileset_test1	1	Valid	Mon Mar 23 09:20:37 2015	nfs-ganesha
filesystem_test2	2	Valid	Mon Mar 23 11:12:59 2015	

2. Use the `mmsnapdir device` command to obtain the name of the snapshot directory for the file system snapshot that you have identified.

In the following example, the fileset snapshot directory is called `.snapshots`.

```
# mmsnapdir fs1
```

Fileset snapshot directory for "fs1" is ".snapshots" (root directory only)

Global snapshot directory for "fs1" is ".snapshots" in root fileset

3. Use the `mmlsfs device -T` command to determine the default mount point of the file system.

In the following example, the default mount point is `/gpfs/fs1`.

```
# mmlsfs fs1 -T
```

flag	value	description
-T	/gpfs/fs1	Default mount point

4. Use the full path to the files and directories that you want to restore and the default mount point that you have determined to obtain the truncated path to the files and directories.

For example:

Full path to the file: `/gpfs/fs1/nfs-ganesha/test1/`

Default mount point: `/gpfs/fs1`

Truncated path: `/nfs-ganesha/test1/`

5. Change the directory to the full snapshot path of the file or the directory to verify.

The full snapshot path is:

```
filesystem_default_mountpoint/snapshot_directory/snapshot_name/truncated_path
```

The full snapshot path using examples in the preceding steps is:

```
/gpfs/fs1/.snapshots/filesystem_test2/nfs-ganesha/test1/
```

6. Do one of the following steps depending on whether you want to restore a file or a directory:

- If you want to restore a file, use the following command:

```
cp -p full_snapshot_path/file_name restore_directory
```

- If you want to restore a directory, change the directory to the *restore_directory* and use the following command:

```
tar -zcf tar_file_name full_snapshot_path/directory_name
```

Restoring a subset of files or directories from a local fileset snapshot

You can restore a subset of files or directories from a local snapshot of an independent fileset in case of accidental deletion.

Ensure the following before you begin:

- You have the full path to the files or directories that you want to restore. The path must include the file system to which these files or directories belong.
- You know which snapshot contains the files or directories that you want to restore.
- You have created a restore directory to which these files or directories are to be restored to avoid accidentally overwriting files or directories.

Use these steps to restore files or directories from a local fileset snapshot.

1. Use the **mm1ssnapshot device** command to list the snapshots in the file system and make a note of the snapshot that contains the files and directories that you want to restore.

device is the name of the file system.

```
# mm1ssnapshot fs1
```

Snapshots in file system fs1:

Directory	SnapId	Status	Created	Fileset
fileset_test1	1	Valid	Mon Mar 23 09:20:37 2015	nfs-ganesha
filesystem_test2	2	Valid	Mon Mar 23 11:12:59 2015	

2. Use the **mmsnapdir device** command to obtain the name of the snapshot directory for the fileset snapshot that you have identified.

In the following example, the fileset snapshot directory is called `.snapshots`.

```
# mmsnapdir fs1
```

```
Fileset snapshot directory for "fs1" is ".snapshots" (root directory only)
Global snapshot directory for "fs1" is ".snapshots" in root fileset
```

3. Use the **mm1sfileset device** command to verify that the fileset status is linked and to determine the full path of the fileset.

In the following example, all filesets are linked and the paths are in the 3rd column.

```
# mm1sfileset fs1
```

Filesets in file system 'fs1':

Name	Status	Path
root	Linked	/gpfs/fs1
nfs-ganesha	Linked	/gpfs/fs1/nfs-ganesha
nfs-ganesha2	Linked	/gpfs/fs1/nfs-ganesha2
nfs-ganesha3	Linked	/gpfs/fs1/nfs-ganesha3
nfs-ganesha4	Linked	/gpfs/fs1/nfs-ganesha4

4. Use the full path to the files and directories that you want to restore and the fileset path that you have determined to obtain the truncated path to the files and directories.

For example:

```
Full path to the file: /gpfs/fs1/nfs-ganesha/test1/
Fileset path:         /gpfs/fs1/nfs-ganesha
Truncated path:      /test1/
```

5. Change the directory to the full snapshot path of the file or the directory to verify.

The full snapshot path is:

```
fileset_path/snapshot_directory/snapshot_name/truncated_path
```

The full snapshot path using examples in the preceding steps is:

```
/gpfs/fs1/nfs-ganesha/.snapshots/fileset_test1/test1/
```

6. Do one of the following steps depending on whether you want to restore a file or a directory:

- If you want to restore a file, use the following command:

```
cp -p full_snapshot_path/file_name restore_directory
```

- If you want to restore a directory, change the directory to the *restore_directory* and use the following command:

```
tar -zcf tar_file_name full_snapshot_path/directory_name
```

Restoring a subset of files or directories from local snapshots using the sample script

You can restore a subset of files or directories from local snapshots using a sample script in case of accidental deletion.

- The **mmcdpsnapqueryrecover** sample script only works on the Linux operating system.

- The sample script retrieves files or directories from all file system and fileset snapshots on the system and presents a list of files that you can choose to restore.
- Regular files are simply copied into the user-specified directory. If the user specifies a directory to be retrieved, the directory is copied into the user-specified directory as a compressed tar file.
- Files and directories that contain spaces in their names can also be retrieved.

Use the **mmcdpsnapqueryrecover** sample script to restore files or directories from snapshots into the user-specified `restorePath` directory as follows.

1. Use the following command to list all copies of a file or directory in a file system or fileset snapshot.

```
/usr/lpp/mmfs/samples/ilm/mmcdpsnapqueryrecover.sh Device \  
--file-path fsPath --destination-dir restorePath
```

Where:

- *device* is the name of the file system.
- *file-path* is the full file path.
- *destination-dir* is the full path of the restore directory.

For example, to get all copies of the file `/gpfs0/gplssnapshot` in the file system `gpfs0` and with `/opt` as the restore directory, enter the following:

```
/usr/lpp/mmfs/samples/ilm/mmcdpsnapqueryrecover.sh /dev/gpfs0 \  
--file-path /gpfs0/gplssnapshot --destination-dir /opt
```

All copies of the specified file are listed as follows:

```
Found regular file in filesystem snapshot: restorFiles1  
1) 5743 Jan 9 08:34 /gpfs0/.snapshots/restorFiles1/gplssnapshot
```

```
Found regular file in filesystem snapshot: restorFiles3  
2) 5882 Jan 9 08:34 /gpfs0/.snapshots/restorFiles3/gplssnapshot
```

```
Found regular file in filesystem snapshot: Restore1  
3) 5886 Jan 14 12:33 /gpfs0/.snapshots/Restore1/gplssnapshot
```

```
Found regular file in filesystem snapshot: Restore2  
4) 5886 Jan 14 12:33 /gpfs0/.snapshots/Restore2/gplssnapshot
```

```
Found regular file in filesystem snapshot: global1  
5) 5886 Jan 14 12:33 /gpfs0/.snapshots/global1/gplssnapshot
```

Which copy of the file/directory (1-5) would you like to restore?

2. From the list, select the file that you want to restore by entering the corresponding number. For example:

```
Which copy of the file/directory (1-5) would you like to restore? 2
```

The copy number 2 is restored to the `/opt` directory.

Chapter 4. Configuring the CES and protocol configuration

After GPFS is configured, Cluster Export Services (CES) and its protocols can be configured, administered, or removed from the system.

Some of the CES and protocol configuration steps might have been completed already through the IBM Spectrum Scale installer. To verify, see the information about IBM Spectrum Scale installer and protocol configuration.

A manual or a minimal installation of CES involves configuration and administrative tasks.

Configuring Cluster Export Services

If you have not configured Cluster Export Services (CES) through the installer, you must configure CES now.

For information on CES features, see the *Implementing Clustered Export Services* topic in the *IBM Spectrum Scale: Advanced Administration Guide*.

Setting up Cluster Export Services shared root file system

If you have not set up a shared root file system through the installer, create one for Cluster Export Services (CES).

The CES shared root (`cesSharedRoot`) is needed for storing CES shared configuration data, protocol recovery, and for some other protocol specific purpose. It is part of the cluster export configuration and is shared between the protocols. Every CES node requires access to the path configured as shared root.

The `mmchconfig` command is used to configure this directory as part of setting up a CES cluster.

The `cesSharedRoot` cannot be changed while any CES nodes are up and running. You need to bring down all CES nodes if you want to modify the shared root configuration.

The `cesSharedRoot` is monitored by the `mmsysmonitor`. If the shared root is not available, the CES node list (`mmces node list`) will show "no-shared-root" and a failover is triggered.

The `cesSharedRoot` cannot be unmounted when the CES cluster is up and running. You need to bring all CES nodes down if you want to unmount `cesSharedRoot` (for example, for doing service action like `fsck`).

To list the current `cesSharedRoot`, run

```
mmisconfig cesSharedRoot
cesSharedRoot /gpfs/gpfs-ces/
```

The recommendation for CES shared root is a dedicated file system (but this is not enforced). It can also be a part (path) of an existing GPFS file system. A dedicated file system can be created with the `mmcrfs` command. In any case, CES shared root must reside on GPFS and must be available when it is configured through `mmchconfig`.

If not already done through the installer, it is recommended that you create a file system for the CES. Some protocol services share information through a cluster-wide file system. It is recommended to use a separate file system for this purpose.

Note: The recommended size for CES shared root file system is greater than or equal to 4GB.

To set up CES, change the configuration to use the new file system:

```
mmchconfig cesSharedRoot=/gpfs/fs0
```

Note: Once GPFS starts back up, by virtue of the fact that the cesSharedRoot is now defined, then CES can be enabled on the cluster.

Configuring Cluster Export Services nodes

If you have not configured Cluster Export Services (CES) nodes through the installer, you must configure them before you configure any protocols.

If not already done during the installation, this must be done before configuring any protocols. Nodes that should participate in the handling of protocol exports need to be configured as CES nodes.

Note: You can have a maximum of 16 nodes in a CES cluster.

For each of the nodes that should handle protocol exports, run:

```
mmchnode -N nodename --ces-enable
```

After configuring all nodes, verify that the list of CES nodes is complete:

```
mmces node list
```

CES nodes may be assigned to CES groups. A CES group is identified by a group name consisting of case-sensitive alphanumeric characters. CES groups may be used to manage CES node and address assignments.

Nodes may be assigned to groups by issuing the following command:

```
mmchnode --ces-group group1 -N node
```

A node may be assigned to multiple groups by issuing the following command:

```
mmchnode --ces-group group1,group2,group3 -N node1,node2
```

The group assignment may also be specified when the node is enabled for CES by issuing the following command:

```
mmchnode --ces-enable --ces-group group1,group2 -N node
```

The node may be removed from a group at any time by issuing the following command:

```
mmchnode --noces-group group1 -N node
```

For more information, see *mmchnode command* in *IBM Spectrum Scale: Administration and Programming Reference*.

Configuring CES protocol service IP addresses

Protocol services are made available through Cluster Export Services (CES) protocol service IP addresses. These addresses are separate from the IP addresses that are used internally by the cluster.

Each CES protocol service IP address is assigned initially to one CES node, either explicitly as specified by the **mmces address add** command, or by the system, but they can be moved later either manually or automatically in response to certain events.

```
mmces address add --ces-node Node1 --ces-ip 192.168.6.6
```

After adding all desired CES protocol service IP addresses, verify the configuration:

```
mmces address list
```

Use `mmces address add --ces-ip 192.168.6.6` to add an IP address to the CES IP address pool. The IP address will be assigned to a CES node according to the CES "Address distribution policy".

CES addresses may be assigned to CES groups. A CES group is identified by a group name consisting of alphanumeric characters which are case-sensitive. Addresses may be assigned to a group when they are defined by issuing the following command:

```
mmces address add --ces-ip 192.168.6.6 --ces-group group1
```

The group assignment may be changed by issuing the following command:

```
mmces address change --ces-ip 192.168.6.6 --ces-group group2
```

The group assignment may be removed by issuing the following command:

```
mmces address change --ces-ip 192.168.6.6 --remove-group
```

A CES address which is associated with a group may only be assigned to a node which is also associated with the same group. A node may belong to multiple groups while an address may not.

As an example, consider a configuration with three nodes -- all of which are able to host addresses on subnet A and two of the nodes are able to host addresses on subnet B. Also four addresses are defined, two on each subnet:

Node1: groups=subnetA,subnetB

Node2: groups=subnetA,subnetB

Node3: groups=subnetA

Address1: subnetA

Address2: subnetA

Address3: subnetB

Address4: subnetB

In this example, Address1 and Address2 may be assigned to any of the three nodes, but Address3 and Address4 may be assigned to only Node1 or Node2.

If an address is assigned to a group for which there are no healthy nodes, the address will remain unassigned until a node in the same group becomes available.

Addresses without a group assignment may be assigned to any node.

For more information, see *mmces command* in *IBM Spectrum Scale: Administration and Programming Reference*

Deploying Cluster Export Services packages on IBM Spectrum Scale existing 4.1.1 and above

Use the following instructions to copy packages onto your protocol nodes and rpm install packages below.

1. Install packages `rpm -ivh <package>` :

SMB: `gpfs.smb-4.2.1_gpfs_27-1.e17.x86_64.rpm`

NFS: `nfs-ganesha-2.2.0-0.2ibm1.e17.x86_64.rpm,nfs-ganesha-utils-2.2.0-02.ibm1e17.x86_64.rpm,nfs-ganesha-gpfs-2.2.0.0.2ibm1.e17.x86_64.rpm`

Object: `spectrum-scale-object python-ldappool`

Performance monitoring: `gpfs.gss.pmcollector-4.1.0-8.e17.x86_64.rpm`, `pmswift-4.1.1-3.noarch.rpm`, `gpfs.gsspsensors-4.1.0-8.e17.x86_64.rpm`

2. Set the server licenses. For each `ces_node`, issue the following command: `mmchlicense server --accept -N ces_node_ips (e.g., mmchlicense server --accept -N 192.168.100.103,192.168.100.104)`
3. Enable CES by issuing the following command: `mmchnode -N ces_nodes --ces-enable (e.g., mmchnode -N 192.168.100.103,192.168.100.104 --ces-enable)`
4. Assign export IPs. For each `export_IP`, issue this command: `mmces address add --ces-ip export_IP`

Verifying the final CES configurations

After you finish the Cluster Export Services (CES) configuration steps, verify the final configuration.

To verify your configuration, run the following command:

```
mm1scluster --ces
```

For more information, see the `mmces node list` and `mmces address list` options in `mmces command` in *IBM Spectrum Scale: Administration and Programming Reference*.

Creating and configuring file systems and filesets for exports

If you have not done so previously, create the file systems and the filesets for the data to be shared through the protocol services. For more information, see `mmcrfs` and `mmcrfileset` in *IBM Spectrum Scale: Administration and Programming Reference*.

Creating a fileset through the GPFS GUI

To create a fileset, log on to the IBM Spectrum Scale GUI and select **Files > Filesets > Create Fileset**.

IBM strongly recommends to configure the file systems to only allow NFSv4 ACLs through the `-k nfs4` option for `mmcrfs`. When using the default configuration profiles (`/usr/lpp/mmfs/profiles`) that are included with IBM Spectrum Scale, the NFSv4 ACL setting is already set from the profile configuration (see “Authorizing file protocol users” on page 200 for details). Also if quotas should be used, enable the quota usage during the file system creation.

For information on unified file and object access, see *Planning for unified file and object access* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Note: Ensure that all GPFS file systems used to export data via NFS are mounted with the `syncnfs` option in order to prevent clients from running into data integrity issues during failover. It is recommended to use the `mmchfs` command to set the `syncnfs` option as default when mounting the GPFS file system.

For more information on creating protocol data exports, see *File system considerations for the NFS protocol* and *Fileset considerations for creating protocol data exports* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Managing protocol services

GPFS provides system administrators with the ability to manage the protocol services.

Configuring and enabling SMB and NFS protocol services

If you have not previously enabled and started the Cluster Export Services (CES) protocol services, enable and start them now.

Prerequisites

When you enable SMB protocol services, the following prerequisites must be met:

- The number of CES nodes must be 16 or lower.
- All CES nodes must be running the same system architecture. For example, mixing nodes based on Intel and Power® is not supported.

When you add new CES nodes to a running system where the SMB protocol is enabled, the following prerequisite must be met:

- All CES nodes must be in SMB HEALTHY state.

When you remove a CES node from a running system where the SMB protocol is enabled, the following prerequisite must be met:

- All CES nodes (except for the node that is being removed) must be in SMB HEALTHY state.

For more information about the SMB states, see **mmces command** in *IBM Spectrum Scale: Administration and Programming Reference*.

Disabling SMB protocol services deletes all configured SMB exports and SMB settings. After re-enabling SMB, all exports and settings must be re-created and reconfigured.

Enabling protocol services

To enable the SMB/NFS services, run one of the following commands:

- ```
mmces service enable SMB
```
- ```
mmces service enable NFS
```

Note: This command starts SMB/NFS on all CES nodes.

GUI navigation

- To enable SMB services in the GUI, log on to the IBM Spectrum Scale GUI and select **Settings > SMB Service**.
- To enable NFS services in the GUI, log on to the IBM Spectrum Scale GUI and select **Settings > NFS Service**.

The protocol services that are used need to be started on all CES nodes:

```
mmces service start SMB -a  
mmces service start NFS -a
```

After you start the protocol services, verify that they are running.

Note: The start and stop are maintenance commands. Stopping a service on a particular protocol node, without first suspending the node ensures the public IP addresses on that node stay with that node. In this event, protocol clients might attempt to connect using these IP addresses and fail to connect to the service.

The sequence of commands to enable file access and then disable file access, using the NFS service as an example, follows:

1. Enable NFS by using the following command:

```
mmces service enable NFS
```

Note: This command also starts NFS on all CES nodes.

Then, you need to set up the authentication before you can add an export. The easiest authentication setup is to use system authentication.

2. Set up the authentication by using userdefined authentication type and file data access method:
`mmuserauth service create --data-access-method file --type userdefined`
3. Add an export by running the following command, where **fs0** is a GPFS file system and **fset0** is an independent fileset:
`mmnfs export add /gpfs/fs0/fset0`
4. Verify that this is configured and running by using the following commands:
`mmces service list -a`
`mmuserauth service list`
`mmnfs export list`
5. Stop NFS and disable NFS protocol on the CES nodes by running the following commands:
`mmces service stop nfs -a`
`mmuserauth service remove --data-access-method file`

Note: Authentication services must be removed for file access before disabling the NFS; and if SMB is enabled and running, then you cannot remove file authentication now.

6. Disable NFS service on the CES nodes by running the following command:
`mmces service disable NFS`

The **mmnfs export change** command currently does not update the export in a dynamic fashion on the running instance of NFS; it stops and starts NFS on all CES nodes.

Note: The NFS configuration is removed when NFS is disabled, so whatever was exported previously is lost. If you want to save the NFS configuration, you should backup the contents of `/var/mmfs/ces/nfs-config/` on any protocol node.

Configuring and enabling the Object protocol service

If you want to use the Cluster Export Services (CES) object service and it was not configured and enabled during the installation, configure object services now.

1. If a file system for the object data has not been created yet, you must create it now (see “mmcrfs command” on page 414).
2. Use the **mmobj** command for the initial configuration of the object stack.

Note: A separate fileset will be created to hold the object data, and data should not be shared with other protocols:

```
mmobj swift base -g /gpfs/fs01 --cluster_hostname clustername --local_keystone --db_password Passw0rd --admin_password Password -i 2000 --enable-s3
```

This example creates a fileset for Object storage in the `/gpfs/fs01` fileset with the specified hostname as access point and 2000 inodes. A local database is created for Keystone authentication and the Amazon S3 emulation is enabled. See “mmobj command” on page 581 for details.

3. After the initial configuration, enable and start the object services by running the following commands:
`mmces service enable OBJ`
`mmces service list -a`
4. Verify that the object service is running as expected:
`mmces service list -a`

Performance tuning for object services

By default, the IBM Spectrum Scale installation sets the number of workers for the object services low. These numbers can be adjusted upwards if you have protocol servers with sufficient cores and memory.

As with most performance tuning, there is no single correct setting. A good starting point for tuning worker counts is to set workers in the `proxy-server.conf` to `auto` so that one worker is started for every core on a protocol node. The other servers can be set to a percentage of the number of cores on your protocol nodes:

- object server set to 75% of core count
- container server set to 50% of core count
- account server set to 25% of core count

Depending on the load of other protocol workloads, the optimal settings for worker count might be higher or lower than this on your system.

For example, if you have 16 cores in your protocol nodes, the following commands can be used to tune your worker settings:

```
mmobj config change --ccrfile proxy-server.conf --section DEFAULT --property workers --value auto
```

```
mmobj config change --ccrfile object-server.conf --section DEFAULT --property workers --value 12
```

```
mmobj config change --ccrfile container-server.conf --section DEFAULT --property workers --value 8
```

```
mmobj config change --ccrfile account-server.conf --section DEFAULT --property workers --value 4
```

Disabling protocol services

If a protocol service is no longer needed, it can be disabled by using the `mmces` command.

To disable a protocol service, enter the appropriate `mmces` command:

- `mmces service disable SMB`

Note: SMB can be removed only when authentication has been cleared (or is user defined).

Disabling SMB services will stop SMB on all CES nodes and remove the SMB configuration from the Cluster Configuration Repository (CCR), the SMB clustered databases (tdb's), and the SMB-related config files in `/var/mmfs/ces` on the CES nodes.

- `mmces service disable NFS`

Note: Disabling NFS services will stop NFS on all CES nodes and remove the NFS configuration from the Cluster Configuration Repository (CCR) and remove `/var/mmfs/ces/nfs-config/` on the CES nodes.

- `mmces service disable OBJ`

Note: For information on disabling the Object services, see the *Understanding and managing the Object services* section in *IBM Spectrum Scale: Advanced Administration Guide*.

Managing protocol user authentication

The system administrator can configure authentication for both object and file access either during the installation of the system or after the installation. If the authentication configuration is not configured during installation, you can manually do it by using the `mmuserauth service create` command from any node in the IBM Spectrum Scale cluster. This section covers the manual method of configuring authentication for file and object access.

Setting up authentication servers to configure protocol user access

Before you start configuring authentication for protocol access, the system administrator needs to ensure that the authentication server is set up properly and the connection between the IBM Spectrum Scale system and authentication server is established properly.

Depending on the requirement, the IBM Spectrum Scale system administrator needs to set up the following servers:

- Microsoft Active Directory (AD) for file and object access
- Lightweight Directory Access Protocol server for file and object access
- Keystone server to configure local, AD, or LDAP-based authentication for object access. Configuring Keystone is a mandatory requirement if you need to have Object access.

AD and LDAP servers are set up externally. You can configure either an internal or external Keystone server. The installation and configuration of an external authentication server must be handled separately. The IBM Spectrum Scale system installation manages the installation and set up of internal Keystone server.

Integrating with AD server

If the authentication method is selected as AD, the customer must set up the AD server before configuring the authentication method in the IBM Spectrum Scale system.

Ensure that you have the following details before you start configuring AD-based authentication:

- IP address or host name of the AD server.
- DNS is configured on all protocol nodes of the system, where the primary DNS needs to be configured as the AD domain controller.
- Domain details such as the following:
 - Domain name and realm.
 - AD admin user ID and password to join the IBM Spectrum Scale system as machine account into the AD domain.
- ID map role of the system is identified.
- Define the ID map range and size depending upon the maximum RID (sum of allocated and expected growth).
- Primary DNS is added in the `/etc/resolv.conf` file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. This is a mandatory requirement when AD is used as the authentication server as the DNS must be able to resolve the host domain and its trusted domains of interest. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, see the corresponding Operating System documentation.
- During the AD join process, a computer account having the same name as the netbios name is searched within the AD domain that will be joined. If the name is not found, a new computer entry is created in the standard location (CN=Computers). If the user chooses to pre-create computer accounts for IBM Spectrum Scale in the AD domain within a particular organizational unit, the computer account must be created with a valid name and it must be passed as the netbios name while configuring the IBM Spectrum Scale system. After the account is created on the AD server, the system must be joined to the AD domain.

To achieve high-availability, you can configure multiple AD domain controllers. While configuring AD-based authentication, you do not need to specify multiple AD servers in the command line to achieve high-availability. The IBM Spectrum Scale system queries the specified AD server for relevant details and configures itself for the AD-based authentication. The IBM Spectrum Scale system relies on the DNS server to identify the set of available AD servers that are currently available in the environment serving the same domain system.

Integrating with LDAP server

If LDAP-based authentication is selected, ensure that the LDAP server is set up with the required schemas to handle the authentication and ID mapping requests. If you need to support SMB data access, LDAP schema must be extended before configuring the authentication.

Ensure that you have the following details before you start configuring LDAP based authentication:

- Domain details such as base dn, and dn prefixes of groups and users, else default values are used. Default user group suffix is <ou=Groups, <base dn> and default user suffix is ou=People, <base dn>.
- IP address or host name of LDAP server.
- Admin user ID and password of LDAP server that is used during LDAP simple bind and for LDAP searches.
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.
- NetBIOS name that is to be assigned for the IBM Spectrum Scale system.
- If you need to have secure communication between the IBM Spectrum Scale system and LDAP, the CA signed certificate that is used by the LDAP server for TLS communication must be placed at the specified location in the system.
- If you are using LDAP with Kerberos, create Kerberos keytab file by using the MIT KDC infrastructure.
- Primary DNS is added in the /etc/resolv.conf file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, refer the corresponding Operating System documentation.

Setting up LDAP server prerequisites:

Before you start configuring the IBM Spectrum Scale system with LDAP server, the following external LDAP server prerequisites must be met:

- The LDAP server must already be configured.
- Enable TLS encryption on the LDAP server, if you need to secure communication between the IBM Spectrum Scale system and LDAP server. Details on configuring SSL or TLS encryption on the server can be obtained from the *OpenLDAP Administrator's Guide*.
- To access SMB shares, LDAP user information must be updated with unique Samba attributes in addition to the attributes that are stored for a normal LDAP user. Ensure that these required Samba attributes are present in the LDAP user entries.
- Ensure you do not have the same user name for different organizational units of the LDAP server that is configured with the IBM Spectrum Scale system.

LDAP bind user requirements:

When an IBM Spectrum Scale system is configured with LDAP as the authentication method, the IBM Spectrum Scale system needs to connect to the LDAP server by using an administrative user ID and password. This administrative user is referred as bind user.

It is recommended that the bind user is given enough privileges that are required by the storage system to mitigate any security concerns.

This bind user must at least have permission to query users and groups that are defined in the LDAP server to allow storage system to authenticate these users. The bind user information (bind dn) is also used by Samba server while making LDAP queries to retrieve required information from the LDAP server.

Note: In the following sections, it is assumed that the user account for the bind user exists in the LDAP directory server. The bind user distinguished name (also known as dn) used in the following examples is uid=ibmbinduser,ou=people,dc=ldapservers,dc=com. This name needs to be updated based on the bind user that is used with the IBM Spectrum Scale system.

OpenLDAP server ACLs:

The OpenLDAP server ACLs define the privileges that are required for the bind user.

The following example uses ACLs that are required for the bind user and other type of users for the sake of completeness. It is likely that a corporate directory server has those ACLs configured already and only the entries for bind user need to be merged correctly in the slapd configuration file (generally, /etc/openldap/slapd.conf file on Linux systems). Follow the ACL ordering rules to ensure that correct ACLs are applied.

```
### some attributes need to be readable so that commands like 'id user',
'getent' etc can answer correctly.
access to attrs=cn,objectClass,entry,homeDirectory,uid,uidNumber,
gidNumber,memberUid
by dn="uid=ibmbinduser,ou=people,dc=ldapservers,dc=com" read

###The following will not list userPassword when ldapsearch is
performed with bind user.

### Anonymous is needed to allow bind to succeed and users to
authenticate, should be
a pre-existing entry already.
access to attrs=userPassword
    by dn="uid=ibmbinduser,ou=people,dc=ldapservers,dc=com" auth
    by self write
    by anonymous auth
    by * none

### Storage system needs to be able to find samba domain account
specified on the mmuserauth service create command.

###It is strongly recommended that domain account is pre-created
to ensure

###consistent access to multiple storage systems.

###Uncomment ONLY if you want storage systems to create domain
account when it does not exist.
#access to dn.base="dc=ldapservers,dc=com"
#    by dn="uid=ibmbinduser,ou=people,dc=ldapservers,dc=com" write
#    by * none

access to dn.regex="sambadomainname=[^,]+,dc=ldapservers,dc=com"
    by dn=" uid=ibmbinduser,ou=people,dc=ldapservers,dc=com" read
    by * none

### all samba attributes need to be readable for samba access
access to attrs=cn,sambaLMPassword,sambaNTPassword,sambaPwdLastSet,
sambaLogonTime,sambaLogoffTime,sambaKickoffTime,sambaPwdCanChange,
sambaPwdMustChange,sambaAcctFlags,displayName,sambaHomePath,
sambaHomeDrive,sambaLogonScript,sambaProfilePath,description,
sambaUserWorkstations,sambaPrimaryGroupSID,sambaDomainName,
sambaMungedDial,sambaBadPasswordCount,sambaBadPasswordTime,
sambaPasswordHistory,sambaLogonHours,sambaSID,sambaSIDList,
sambaTrustFlags,sambaGroupType,sambaNextRid,sambaNextGroupRid,
sambaNextUserRid,sambaAlgorithmicRidBase,sambaShareName,
sambaOptionName,sambaBoolOption,sambaIntegerOption,
```

```
sambaStringOption,sambaStringListoption
  by dn="uid=ibmbinduser,ou=people,dc=ldapservers,dc=com" read
  by self read
  by * none
```

```
### Need write access to record bad failed login attempt
access to attrs=cn,sambaBadPasswordCount,sambaBadPasswordTime,
sambaAcctFlags by dn="uid=ibmbinduser,ou=people,dc=ldapservers,
dc=com" write
```

```
### Required to check samba schema
access to dn.base=* by dn="uid=ibmbinduser,ou=people,
dc=ldapservers,dc=com" read
```

Tivoli Directory Server ACLs:

The Tivoli Directory Server ACLs define the privileges that are required for the bind user, when using Tivoli Directory Server.

These ACLs are provided in the LDIF format and can be applied by submitting the **ldapmodify** command.

```
dn: dc=ldapservers,dc=com
changetype: modify
```

```
add: ibm-filterAclEntry
```

```
ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapservers,dc=com:
(objectClass=sambaSamAccount):normal:rsc:sensitive:rsc:critical:rsc
```

```
-
```

```
add: ibm-filterAclEntry
```

```
ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapservers,dc=com:
(objectClass=sambaDomain):normal:rws:c:sensitive:rws:c:critical:rws:c
```

```
dn:uid=ibmbinduser,ou=people,dc=ldapservers,dc=com
```

```
add:aclEntry
```

```
aclentry: access-id:uid=ibmbinduser,ou=people,dc=ldapservers,dc=com:at.cn:r:at.
objectClass:r:at.homeDirectory:r:at.uid:r:at.uidNumber:s:
```

```
at.gidNumber:r:at.memberUid:r:at.userPassword:sc:at.sambaLMPassword:r:at.
sambaNTPassword:r:at.sambaPwdLastSet:r:at.sambaLogonTime:r:
```

```
at.sambaLogoffTime:r:at.sambaKickoffTime:r:at.sambaPwdCanChange:r:at.
sambaPwdMustChange:r:at.sambaAcctFlags:r:at.displayName:r:
```

```
at.sambaHomePath:r:at.sambaHomeDrive:r:at.sambaLogonScript:r:at.sambaProfilePath:
r:at.description:r:at.sambaUserWorkstations:r:
```

```
at.sambaPrimaryGroupSID:r:at.sambaDomainName:r:at.sambaMungedDial:r:at.
sambaBadPasswordCount:r:at.sambaBadPasswordTime:r:
at.sambaPasswordHistory:r:at.sambaLogonHours:r:at.sambaSID:r:at.sambaSIDList:r:at.
sambaTrustFlags:r:at.sambaGroupType:r:
at.sambaNextRid:r:at.sambaNextGroupRid:r:at.sambaNextUserRid:r:at.
sambaAlgorithmicRidBase:r:at.sambaShareName:r:at.sambaOptionName:r:
```

```
at.sambaBoolOption:r:at.sambaIntegerOption:r:at.sambaStringOption:r:at.
sambaStringListoption:r:at.sambaBadPasswordCount:rws:c:
```

```
at.sambaBadPasswordTime:rws:c:at.sambaAcctFlags:rws:c
```

```
### Storage system needs to be able to find samba domain account specified
on the mmuserauth service create command.

###It is strongly recommended that domain account is pre-created to ensure

###consistent access to multiple storage systems.

###Uncomment ONLY if you want storage systems to create domain account when
it does not exist.
```

```
dn: dc=ldapservers,dc=com

changetype: modify

add:ibm-filterAclEntry

ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapservers,
dc=com:(objectclass=domain):object:grant:a
```

See *IBM Tivoli Directory Server Administration Guide* for information about applying these ACLs on the Tivoli Directory Server.

Updating LDAP user information with Samba attributes:

If you need to support SMB data access, LDAP schema must be extended to store more attributes such as SID, Windows password hash to the POSIX user object. To use Samba accounts, update LDAP user information with unique Samba attributes.

The following sample LDIF file shows the minimum required samba attributes:

```
dn: cn=SMBuser,ou=People,dc=ibm,dc=com
changetype: modify
add : objectClass
objectClass: sambaSamAccount
-
add: sambaSID
sambaSID: S-1-5-21-1528920847-3529959213-2931869277-1102
-
add:sambaPasswordHistory
sambaPasswordHistory: 0000000000000000000000000000000000000000000000000000000000000000
-
add:sambaNTPassword
sambaNTPassword: (valid samba password hash )
-
add:sambaPwdLastSet
sambaPwdLastSet: 1263386096
-
add:SambaAcctFlags
sambaAcctFlags: [U          ]
```

Note: Attributes must be separated with a dash as the first and only character on a separate line.

Perform the following steps to create the values for sambaNTPassword, sambaPwdLastSet, and SambaAcctFlags, which must be generated from a PERL module:

1. Download the module from <http://search.cpan.org/~bjkuit/Crypt-SmbHash-0.12/SmbHash.pm>. Create and install the module by following the readme file.
2. Use the following PERL script to generate the LM and NT password hashes:

```
# cat /tmp/Crypt-SmbHash-0.12/gen_hash.pl
#!/usr/local/bin/perl
use Crypt::SmbHash;
```



```

$username = $ARGV[0];
$password = $ARGV[1];
if ( !$password ) {
    print "Not enough arguments\n";
    print "Usage: $0 username password\n";
    exit 1;
}
$suid = (getpwnam($username))[2];
my ($login,undef,$uid) = getpwnam($ARGV[0]);
ntlmgen $password, $lm, $nt;
printf "%s:%d:%s:%s:[%-11s]:LCT-%08X\n", $login, $uid, $lm, $nt, "U", time;

```

3. Generate the password hashes for any user as in the following example for the user test01:

```

# perl gen_hash.pl SMBuser test01

:0:47F9DBCCD37D6B40AAD3B435B51404EE:82E6D500C194BA5B9716495691FB7DD6:
[U          ]:LCT-4C18B9FC

```

Note: The output contains login name, uid, LM hash, NT hash, flags, and time, with each field separated from the next by a colon. The login name and uid are omitted because the command was not run on the LDAP server.

4. Use the information from step 3 to update the LDIF file in the format that is provided in the example at the beginning of this topic.
 - To generate the sambaPwLastSet value, use the hexadecimal time value from step 3 after the dash character and convert it into decimal.
 - A valid samba SID is required for a user to enable that user's access to an IBM Spectrum Scale share. To generate the samba SID, multiply the user's UID by 2 and add 1000. The user's SID must contain the samba SID from the sambaDomainName, which is either generated or picked up from the LDAP server, if it exists. The following attributes for sambaDomainName LDIF entry are required:

```

dn: sambaDomainName=(IBM Spectrum Scale system),dc=ibm,dc=com
sambaDomainName: (IBM Spectrum Scale system name)
sambaSID: S-1-5-21-1528920847-3529959213-2931869277
sambaPwHistoryLength: 0
sambaMaxPwAge: -1
sambaMinPwAge: 0

```

This entry can be created by the LDAP server administrator by using either of the following two methods:

- Write and run a bash script similar to the following example:

```

sambaSID=
  for num in 1 2 3 ;do
    randNum=$(od -vAn -N4 -tu4 < /dev/urandom | sed -e 's/ //g')
    if [ -z "$sambaSID" ];then
      sambaSID="S-1-5-21-$randNum"
    else
      sambaSID="${sambaSID}-$ {randNum}"
    fi
  done
echo $sambaSID

```

Then, use the samba SID generated to create the LDIF file. The sambaDomainName must match the IBM Spectrum Scale system name.

- When you run the **mmuserauth service create** command, the system creates the sambaDomainName, if it does not exist.

The sambaSID for every user must have the following format: (samba SID for the domain)-(userID*2+1000). For example: S-1-5-21-1528920847-3529959213-2931869277-1102

Note: To enable access to more than one IBM Spectrum Scale system, the domain SID prefix of all of the systems must match. If you change the domain SID for an IBM Spectrum Scale system on the LDAP server, you must restart CTDB on that IBM Spectrum Scale system for the change to take effect.

5. Submit the `ldapmodify` command as shown in the following example to update the user's information :

```
# ldapmodify -h localhost -D cn=Manager,dc=ibm,dc=com -W -x -f /tmp/samba_user.ldif
```

Integrating with Keystone Identity Service

You need to use either an external or an internal Keystone server with the IBM Spectrum Scale system to configure an authentication method for object users. This Keystone server works with an external authentication server such as AD or LDAP or with a local database to resolve the authentication requests.

Before you configure authentication for object, ensure that the object services are enabled. To enable object services, use the `mmces service enable obj` command.

Prerequisites

Ensure that you have the following details before you start configuring local authentication for object access:

- Keystone host name must be defined and configured on all protocol nodes of the cluster. This host name returns one of the CES IP addresses, such as a round-robin DNS. It could also be a fixed IP of a load balancer that distributes requests to one of the CES nodes. This host name is also used to create the Keystone endpoints.
- If you want the Keystone server to use external CA signed certificate for signing the keystone generated tokens, ensure that the following certificates are available in `/var/mmfs/tmp` directory of the node where you run the commands:
 - `/var/mmfs/tmp/signing_cert.pem`
 - `/var/mmfs/tmp/signing_key.pem`
 - `/var/mmfs/tmp/signing_cacert.pem`

Note: By default, the system uses internal self-signed certificate that is generated by `keystone-manage pki_setup`.

- If you want to enable SSL on Keystone, ensure that the following certificates that are placed on the `/var/mmfs/tmp` directory of the node where commands are run:
 - `/var/mmfs/tmp/ssl_cert.pem`
 - `/var/mmfs/tmp/ssl_key.pem`
 - `/var/mmfs/tmp/ssl_cacert.pem`
 - `/var/mmfs/tmp/ks_ext_cacert.pem`

Note: If you are not using external Keystone server, the IBM Spectrum Scale installation process by default configures the object authentication with local authentication as the authentication method.

Configuring authentication and ID mapping for file access

The system administrator can decide whether to configure authentication and ID mapping method either during the installation of the IBM Spectrum Scale system or after the installation. If the authentication configuration is not configured during installation, you can manually do it by using the `mmuserauth service create` command from any protocol node of the IBM Spectrum Scale system. This section covers the manual method of configuring authentication for file access.

You can configure the following external authentication servers for file access:

- Active Directory (AD)
- Light Weight Directory Access Protocol (LDAP)
- Network Information Service (NIS)

Ensure that the following RPMs are installed on all protocol nodes before you start configuring the authentication method:

- **For AD:**
 - * bind-utils
- **For LDAP:**
 - * openldap-clients
 - * sssd and its dependencies (particularly sssd-common and sssd-ldap)
- **For NIS:**
 - * sssd and its dependencies (particularly sssd-common and sssd-proxy)
 - * ypbind and its dependencies (yp-tools)

AD-based authentication for file access

You can configure Microsoft Active Directory (AD) as the authentication server to manage the authentication requests and to store user credentials.

You can configure AD-based authentication with the following ID mapping methods:

- RFC2307
- Automatic
- LDAP

RFC2307 ID mapping

In the RFC2307 ID mapping method, the user and group IDs are stored and managed in the AD server and these IDs are used by the IBM Spectrum Scale system during file access. The RFC2307 ID mapping method is used when you want to have multiprotocol access. That is, you can have both NFS and SMB access over the same data.

Automatic ID mapping

In the automatic ID mapping method, user ID and group ID are automatically generated and stored within the IBM Spectrum Scale system. When an external ID mapping server is not present in the environment or cannot be used, then this ID mapping method can be used. This method is typically used if you have SMB only access and do not plan to deploy multiprotocol access. That is, the AD-based authentication with automatic ID mapping is not used if you need to allow NFS and SMB access to the same data.

LDAP ID mapping

In the LDAP mapping method, user ID and group ID are stored and managed in the LDAP server, and these IDs are used by the IBM Spectrum Scale system during file access. The LDAP ID mapping method is used when you want to have multiprotocol access. That is, you can have both NFS and SMB access over the same data.

Setting up a range of ID maps that can be allotted to the users:

You can optionally specify the pool of values from which the UIDs and GIDs are assigned by the IBM Spectrum Scale system to Active Directory users and groups. When a user or group is defined in AD, it is identified by a security identifier (SID), which includes a component that is called Relative Identifier (RID). The RID value depends on the number of users and groups in the Active Directory domain. The **--idmap-range** and **--idmap-range-size** parameters of the **mmuserauth service create** command specify the pool from which UIDs and GIDs are assigned by the IBM Spectrum Scale system to AD users and group of users.

The ID map range is defined between a minimum and maximum value. The default minimum value is 10000000 and the default maximum value is 299999999, and the default range size is 1000000. This allows for a maximum of 290 unique Active Directory domains.

The ID map range size specifies the total number of UIDs and GIDs that are assignable per domain. For example, if range is defined as 10000-20000, and range size is defined as 2000 (--idmap-range 10000-20000:2000), five domains can be mapped, each consisting of 2000 IDs. Ensure that range size is defined such that at least three domains can be mapped. The range size is identical for all AD domains that are configured by the IBM Spectrum Scale system. Choose an ID map range size that allows for the highest anticipated RID value among all of the anticipated AD users and group of users in all of the anticipated AD domains. Ensure that the range size value, when originally defined, takes into account the planned growth in the number of AD users and groups of users. The ID map range size cannot be changed after the IBM Spectrum Scale system is configured with Active Directory as the authentication server.

Whenever a user or user group from an AD domain accesses the IBM Spectrum Scale system, a range is allocated per domain. UID or GID for a user or user group is allocated depending upon this range and the RID of the user or user group. If RID of any user or group is greater than the range size, then that user or user group is mapped into extension ranges depending upon the number of available ranges. If the number of ranges (default value is 290) runs out, then mapping requests for a new user or user group (or new extension ranges for user and group that is already known) are ignored and thus that user and user group cannot access the data.

Choosing range size

1. Determine the highest Active Directory RID that is currently assigned. You can use the **dcdiag** command at the command prompt of the operating system of the server that is hosting Active Directory to determine the value of the **rIDNextRID** attribute. For example:

```
# dcdiag /s:IP_of_system_hosting_AD /v /test:ridmanager
```

Specifically,

```
C:\Program Files\Support Tools>dcdiag /s:10.0.0.123 /v /test:ridmanager
```

The following output is displayed:

```
Starting test: RidManager
* Available RID Pool for the Domain is 1600 to 1073741823
* win2k8.pollux.com is the RID Master
* DsBind with RID Master was succesRFC23071
* rIDAllocationPool is 1100 to 1599
* rIDPreviousAllocationPool is 1100 to 1599
* rIDNextRID: 1174
```

In this example, the **rIDNextRID** value is 1174. Another way to determine the current value for **rIDNextRID** is to run an LDAP query on the following DN Path:

```
CN=Rid Set,Cn=computername,ou=domain controllers,DC=domain,DC=COM
```

If there is more than one domain controller serving the Active Directory domain, determine the highest RID among all of the domain controllers. Similarly, if there is more than one domain, determine the highest RID among all of the domains.

2. Estimate the expected number of users and groups that might be added in future, in addition to the current number of users and groups.
3. Add the highest RID determined in step 1 to the number of users and groups that were estimated in the previous step. The result is the estimate for the value of the range size.

Considerations for changing the ID map range and range size:

If the IBM Spectrum Scale system is configured to use AD-based authentication, only the maximum value of ID map range can be changed. All other changes to ID map range and size, except increasing the maximum value of ID map range require reconfiguration of authentication, which results in loss of access

to data. For example, if you used the `--idmap-range` as 3000-10000 and `--idmap-range-size` as 2000, you can increase only the value 10000 to accommodate more users per domain, without having an impact on the access to the data.

To change the ID mapping of an existing AD-based authentication configuration, either to change the range minimum value, decrease the range maximum value, or change the range size, you must complete the following steps:

Note: The `mmuserauth service remove` command results in loss of access.

1. Submit the `mmuserauth service remove` command and do not specify the `--idmapdelete` option.
2. Submit the `mmuserauth service remove` command and do specify the `--idmapdelete` option.
3. Submit the `mmuserauth service create` command with the options and values that you want for the new Active Directory configuration.

Important: If you do not perform the preceding three steps in sequence, results are unpredictable and can include complete loss of data access.

Prerequisite for configuring AD-based authentication for file access:

See “Integrating with AD server” on page 70 for more information on the prerequisites for integrating AD server with the IBM Spectrum Scale system.

You need to run the `mmuserauth service create` command with the following mandatory parameters to create AD based authentication for file access:

- `--type ad`
- `--data-access-method file`
- `--servers <comma-delimited server host names or IP addresses>`
- `--netbios-name <netBiosName>`
- `--user-name <admin-username>`
- `--password <admin-password>`. This is optional while entering the parameters but the system prompts you to enter the password when you run the command.
- `--unixmap-domains <unixDomainMap>`. This option is mandatory if RFC2307 ID mapping is used. For example, `--unixmap-domains DOMAINS(5000-20000)`. Specifies the Active Directory domains for which user ID and group ID should be fetched from the Active directory server (RFC2307 schema attributes)
- `--idmap-role master | subordinate`. While using automatic ID mapping, in order to have same ID maps on systems sharing Active File Manager (AFM) relationship, you need to export the ID mappings from the system whose ID map role is master to the system whose ID map role is subordinate.

See the `mmuserauth service create` command for more information on each parameter.

Prerequisites for configuring AD with RFC2307

The following prerequisites are specific to AD with RFC2307 configuration:

- RFC2307 schema is extended on the AD and all UNIX attributes (including UID and GID) are populated.
- If a trusted domain is configured with ID mapping from RFC2307, the trusted domain must have two-way trust with the host domain, which is the Active Directory domain that is configured for use with the IBM Spectrum Scale system. For example, assume that there are three domains in trusted relationship, X, Y, Z, and that the IBM Spectrum Scale system is configured with domain X as the host domain. If RFC2307 ID mappings are required for domains Y and Z, domains Y and Z must each have a two-way trust with the domain X. $X \leftrightarrow Y$; $X \leftrightarrow Z$.
- User and group in the Active Directory domain, configured with ID mapping from RFC2307, must have a valid UID and a valid GID assigned to enable access to IBM Spectrum Scale system exports.

The UID and GID number that is assigned must be within the ID map range that is specified in the **mmuserauth service create** command. Any users or groups from this domain that do not have UID or GID attributes configured are denied access.

Note: The primary Windows group that is assigned to an AD user must have a valid GID assigned within the specified ID mapping range. The primary Windows group is usually located in the Member Of tab in the user's properties. The primary Windows group is different from the UNIX primary group, which is listed in the UNIX Attributes tab. A user is denied access if that user's Windows primary group does not have a valid GID assigned. The UNIX primary group attribute is ignored.

In case of a mutual trust setup between two independent AD domains, DNS forwarding must be configured between the two trust.

Configuring AD-based authentication with automatic ID mapping:

When the IBM Spectrum Scale system is configured for AD-based authentication, automatic ID mapping method can be used to create UID or GID of a user or group respectively. The ID maps are stored within the IBM Spectrum Scale system.

The following provides an example of how to configure an IBM Spectrum Scale system with Active Directory and automatic ID mapping.

1. Issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ad --data-access-method file --netbios-name
ess --user-name administrator --idmap-role master --servers myADserver
--password PasswOrd --idmap-range-size 1000000 --idmap-range 10000000-299999999
```

The system displays the following output:

File Authentication configuration completed successfully.

2. Verify the authentication configuration by issuing the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS                VALUES
-----
ENABLE_NFS_KERBEROS      false
SERVERS                  myADserver
USER_NAME                administrator
NETBIOS_NAME             ess
IDMAP_ROLE               master
IDMAP_RANGE              10000000-299999999
IDMAP_RANGE_SIZE         1000000
UNIXMAP_DOMAINS          none

OBJECT access not configured
PARAMETERS                VALUES
-----
```

3. Verify the user resolution on the system:

```
# id "DOMAIN\user1"
uid=12001172(DOMAIN\user1) gid=12001174(DOMAIN\group1) groups=12001174
(DOMAIN\group1),12001172(DOMAIN\user1),12000513(DOMAIN\domain users),
11000545(BUILTIN\users)
```

Prerequisite for configuring Kerberos-based SMB access:

The following requirements must be met to configure IBM Spectrum Scale for Kerberized SMB access:

- The time must be synchronized across the KDC server, the IBM Spectrum Scale cluster, and the SMB clients, or else access to an SMB share could be denied.
- In MIT KDC configurations for the SMB services, the service principal name must use the NetBIOS name and the realm name. For example, if the NetBIOS name is FOO and the realm is KDC.COM, the service principal name should be cifs/foo@KDC.COM. The NetBIOS name is the value specified for the option `--netbios_name` in the `mmuserauth` command. The realm may be discovered from the value stored for `Alt_Name` returned from the command: `wbinfo -D <domain>`.
- The clients should use only the NetBIOS name when accessing an SMB share. Using any other name or IP address might either cause a failure to connect or fallback to NTLM authentication.
- With Active Directory KDC, you can use DNS alias (CNAME) for Kerberized SMB access. To use the alias, you must register the DNS alias (CNAME) record for the NetBIOS name (system account name) using the `SetSPN` tool available on Active Directory server. For example, if the NetBIOS name is FOO and the DNS alias is BAR, use the `SetSPN` tool from the command prompt of the Active Directory server to register the record, "setspn -A cifs/BAR FOO". Not registering the DNS alias record for the NetBIOS name might cause access to the SMB shares to be denied with the error code, `KDC_ERR_S_SPRINCIPAL_UNKNOWN`.
- On Linux clients, to use Kerberized SMB access for IBM Spectrum Scale configured with MIT KDC, you must at least have the 3.5.9 version of Samba client installed. The Linux clients having an older Samba client might encounter the following error, while trying to access SMB shares:

```
ads_krb5_mk_req: krb5_get_credentials failed for foo$@KDC.COM (Server not found in Kerberos database)
cli_session_setup_kerberos: spnego_gen_negTokenTarg failed: Server not found in Kerberos database
```

To determine if a client has authenticated via Kerberos, either verify at the client or look for the following in the `/var/adm/ras/log.smbd` log file on IBM Spectrum Scale.

The Kerberos ticket principal name is [`<username>@domain`].

Note that Samba log level must be set to at least 3 for the authentication to be logged.

To adjust the Samba log level, issue the command:

```
/usr/lpp/mmfs/bin/net conf setparm global 'log level' 3
```

Note: It is not recommended to run for extended periods of time at log levels higher than 1 as this could impact performance.

Configuring AD-based authentication with RFC2307 ID mapping:

You can configure IBM Spectrum Scale system authentication with Active Directory (AD) and RFC2307 and Active Directory (AD) with Kerberos and RFC2307 ID mapping. In these authentication methods, use Active Directory to store user credentials and RFC2307 server to store UIDs and GIDs. This is useful when you are planning to use any pre-existing UNIX client or NFS and SMB protocols for data access with the AFM feature of the IBM Spectrum Scale system. If you use AD-based authentication and the ID maps are not configured with RFC2307, the IBM Spectrum Scale system uses the automatic ID mappings by default.

The following provides an example of configuring AD with RFC2307 ID mapping:

1. Submit the `mmuserauth service create` command as shown in the following example:

```
# mmuserauth service create --type ad --data-access-method file --netbios-name
ess --user-name administrator --idmap-role master --servers myADserver
--password Password --idmap-range-size 1000000 --idmap-range 10000000-299999999
--unixmap-domains 'DOMAIN(5000-20000)'
```

The system displays the following output:

```
File authentication configuration completed successfully.
```

2. Issue the `mmuserauth service list` to verify the authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS                VALUES
-----
ENABLE_NFS_KERBEROS      false
SERVERS                  myADserver
USER_NAME                administrator
NETBIOS_NAME             ess
IDMAP_ROLE               master
IDMAP_RANGE              10000000-299999999
IDMAP_RANGE_SIZE         1000000
UNIXMAP_DOMAINS          DOMAIN(5000-20000)
LDAPMAP_DOMAINS          none
```

```
OBJECT access not configured
PARAMETERS                VALUES
```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from RFC2307 attributes on the AD server.

```
# id DOMAIN\administrator
uid=10002(DOMAIN\administrator) gid=10000(DOMAIN\domain users)
groups=10000(DOMAIN\domain users)
```

Configuring AD using Kerberos with RFC2307 ID mapping:

1. Submit the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --data-access-method file --type ad --netbios-name
kknode_v42 --servers myADserver --user-name administrator --password Passw0rd --idmap-role master
--enable-nfs-kerberos --unixmap-domains "DOMAIN(10000-200000)"
```

The system displays the following output:

```
File authentication configuration completed successfully.
```

2. Issue the **mmuserauth service list** to verify the authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS                VALUES
-----
ENABLE_NFS_KERBEROS      true
SERVERS                  myADserver
USER_NAME                administrator
NETBIOS_NAME             kknode_v42
IDMAP_ROLE               master
IDMAP_RANGE              10000000-299999999
IDMAP_RANGE_SIZE         1000000
UNIXMAP_DOMAINS          DOMAIN(1000-200000)
LDAPMAP_DOMAINS          none
```

```
OBJECT access not configured
PARAMETERS                VALUES
```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from RFC2307 attributes on the AD server.

```
# id DOMAIN\administrator
uid=10002(DOMAIN\administrator) gid=40000(DOMAIN\domain users)
groups=11000545(BUILTIN\users),11000544 (BUILTIN\administrators)
```

Best practices for configuring AD with RFC2307 as the authentication method:

It is recommended to adhere to the following best practices if you configure AD with RFC2307 as the authentication method:

- Remove any internal ID mappings present in the system before you configure AD with RFC2307. Otherwise, the system might detect the internal ID mappings instead of the RFC2307 ID mapping and abort the operation with an error message. In such situations, you are expected to clean up the entire authentication and ID mapping by using the **mmuserauth service remove** and **mmuserauth service remove --idmapdelete** command and then reconfigure AD authentication and RFC2307 ID mapping.
- If data is already present on the system, a complete removal of the authentication and ID mapping can cause permanent loss of data access.
- Using UIDs and GIDs greater than 1000 can avoid an overlap of IDs used by end users, administrative users, and operating system component users of the IBM Spectrum Scale system.

You can use AD-based authentication and RFC2307 ID mapping if you want to use the AFM feature of the IBM Spectrum Scale system.

Limitations of the mmuserauth service create command while configuring AD with RFC2307:

The **mmuserauth service create** command that is used to configure authentication has the following limitations:

- The **mmuserauth service create** command does not check the two-way trust between the host domain and the RFC2307 domain that is required for ID mapping services to function properly. The customer is responsible for configuring the two-way trust relationship between these domains.
- The customer is responsible for installing RFC2307 on the desired AD server, and for assigning UIDs to users and GIDs to groups. The command does not return an error if RFC2307 is not installed, or if a UID or GID is not assigned.

Configuring AD-based authentication with LDAP ID mapping:

AD authentication with LDAP ID mapping provides a way for IBM Spectrum Scale to read ID mappings from an LDAP server as defined in RFC 2307. The LDAP server must be a stand-alone LDAP server. Mappings must be provided in advance by the administrator by creating the user accounts in the AD server and the `posixAccount` and `posixGroup` objects in the LDAP server. The names in the AD server and in the LDAP server have to be the same. This ID mapping approach allows the continued use of existing LDAP authentication servers that store records in the RFC2307 format. The group memberships defined in the AD server are also be honored in the system.

In the following example, AD is configured with LDAP ID mapping.

1. Submit the **mmuserauth service create** command as shown in the following example:

```
mmuserauth service create --data-access-method file --type ad --servers myADserver
--user-name administrator --password Passw0rd --netbios-name specscale
--idmap-role master --ldapmap-domains "DOMAIN1(type=stand-alone:range=1000-100000
:ldap_srv=myLDAPserver:usr_dn=ou=People,dc=example,dc=com:grp_dn=ou=Groups,dc=example,
dc=com:bind_dn=cn=manager,dc=example,dc=com:bind_dn_pwd=password)
```

Note: The `bind_dn_password` cannot contain the following special characters: semicolon (;), colon (:), opening brace ('), or closing brace (').

The system displays the following output:

```
File authentication configuration completed successfully.
```

2. Issue the **mmuserauth service list** to verify the authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS VALUES
-----
ENABLE_NFS_KERBEROS false
```

```

SERVERS myADserver
USER_NAME administrator
NETBIOS_NAME specscale
IDMAP_ROLE master
IDMAP_RANGE 10000000-299999999
IDMAP_RANGE_SIZE 1000000
UNIXMAP_DOMAINS none
LDAPMAP_DOMAINS DOMAIN1(type=stand-alone: range=1000-100000:
ldap_srv=myLDAPserver:usr_dn=ou=People,dc=example,dc=com:
grp_dn=ou=Groups,dc=example,dc=com:bind_dn=cn-manager,dc=example,dc=com)

```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from LDAP attributes on the AD server.

```

# id DOMAIN\administrator
uid=10002(DOMAIN\administrator) gid=10000(DOMAIN\domain users)
groups=10000(DOMAIN\domain users)

```

Configuring LDAP-based authentication for file access

Using LDAP-based authentication can be useful when you use an external LDAP server to store user information and user passwords. In this authentication method, you can use LDAP as the authentication as well as the ID mapping server for both NFS and SMB. Appropriate SMB schema needs to be uploaded in the LDAP if you plan to have SMB access.

Based on the level of security, the following configurations are possible:

- LDAP with TLS
- LDAP with Kerberos
- LDAP with TLS and Kerberos
- LDAP

Using LDAP with TLS secures the communication between the IBM Spectrum Scale system and the LDAP server, assuming that the LDAP server is configured for TLS.

You can use LDAP with Kerberos for higher security reasons. Kerberos is a network authentication protocol that provides secured communication by ensuring passwords are not sent over the network to the system. LDAP with Kerberos is typically used where an MIT KDC infrastructure exists and you are using it for various Kerberized application or if you want to have NFS and SMB with Kerberized access for higher security reasons.

The LDAP server might need to handle the login requests and ID mapping requests from the client that uses SMB protocol. Usually, the ID mapping requests are cached and they do not contribute to the load on the LDAP server unless the ID mapping cache is cleared due to a maintenance action. If the LDAP server cannot handle the load or a high number of connections, then the response to the login requests is slow or it might time out. In such cases, users need to retry their login requests.

It is assumed that LDAP server is set up with the required schemas installed in it to handle the authentication and ID mapping requests. If you need to support SMB data access, LDAP schema must be extended to enable storing of additional attributes such as SID, Windows password hash to the POSIX user object.

Note: The IBM Spectrum Scale system must not be configured with any authentication method before using LDAP as the authentication system for file access.

See “Integrating with LDAP server” on page 71 for more information on the prerequisites for integrating LDAP server with the IBM Spectrum Scale system.

Configuring LDAP with TLS for file access:

You can configure LDAP with TLS as the authentication method for file access. Using TLS with LDAP helps you to have a secure communication channel between the IBM Spectrum Scale system and LDAP server.

In the following example, LDAP is configured with TLS as the authentication method for file access.

1. Ensure that the CA certificate for LDAP server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem`; specifically, on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```
# stat /var/mmfs/tmp/ldap_cacert.pem
File: /var/mmfs/tmp/ldap_cacert.pem
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. Issue the `mmuserauth service create` command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-server-tls
```

The system displays the following output:

```
File authentication configuration completed successfully.
```

3. Issue the `mmuserauth service list` command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS                VALUES
-----
ENABLE_SERVER_TLS         true
ENABLE_KERBEROS           false
USER_NAME                 cn=manager,dc=example,dc=com
SERVERS                   myLDAPserver
NETBIOS_NAME              ess
BASE_DN                   dc=example,dc=com
USER_DN                   none
GROUP_DN                  none
NETGROUP_DN              none
USER_OBJECTCLASS          posixAccount
GROUP_OBJECTCLASS         posixGroup
USER_NAME_ATTRIB         cn
USER_ID_ATTRIB            uid
KERBEROS_SERVER           none
KERBEROS_REALM            none

OBJECT access not configured
PARAMETERS                VALUES
-----
```

4. Verify the user resolution on system present in LDAP:

```
# id ldapuser2
uid=1001(ldapuser2) gid=1001(ldapuser2) groups=1001(ldapuser2)
```

Configuring LDAP with Kerberos for file access:

You can configure LDAP with Kerberos as the authentication method for file access. Using Kerberos with LDAP provides more security for the communication channel between the IBM Spectrum Scale system and LDAP server.

Example for configuring LDAP with Kerberos as the authentication method for file access.

1. Ensure that the keytab file is also placed under the `/var/mmfs/tmp` directory with the name as `krb5.keytab` on the node where the command is run. Perform validation of keytab file availability with desired name at required location:

```
# stat /var/mmfs/tmp/krb5.keytab
File: /var/mmfs/tmp/krb5.keytab
Size: 502 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169898 Links: 1
Access: (0600/-rw-----) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 14:31:18.244837381 +0530
Modify: 2015-01-23 12:45:05.475837381 +0530
Change: 2015-01-23 12:45:05.476837381 +0530
Birth: -
```

2. Issue the `mmuserauth service create` command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-kerberos --kerberos-server myKerberosServer
--kerberos-realm example.com
```

The system displays the following output:

File authentication configuration completed successfully.

3. Issue the `mmuserauth service list` command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS                VALUES
-----
ENABLE_SERVER_TLS         false
ENABLE_KERBEROS           true
USER_NAME                  cn=manager,dc=example,dc=com
SERVERS                    myLDAPserver
NETBIOS_NAME               ess
BASE_DN                    dc=example,dc=com
USER_DN                    none
GROUP_DN                   none
NETGROUP_DN                none
USER_OBJECTCLASS           posixAccount
GROUP_OBJECTCLASS          posixGroup
USER_NAME_ATTRIB           cn
USER_ID_ATTRIB             uid
KERBEROS_SERVER            myKerberosServer
KERBEROS_REALM             example.com
```

```
OBJECT access not configured
PARAMETERS                VALUES
-----
```

Configuring LDAP with TLS and Kerberos for file access:

You can configure LDAP with TLS and Kerberos as the authentication method for file access. Using Kerberos and TLS with LDAP provides maximum security for the communication channel between the IBM Spectrum Scale system and the LDAP server.

Provides an example on how to configure LDAP with TLS and Kerberos as the authentication method for file access.

1. Ensure that the CA certificate for LDAP server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem`; specifically, on the protocol node where the command is run. Perform validation of CA cert availability with desired name at the required location as shown in the following example:

```
# stat /var/mmfs/tmp/ldap_cacert.pem
File: /var/mmfs/tmp/ldap_cacert.pem
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. Ensure that the keytab file is placed under `/var/mmfs/tmp` directory name as `krb5.keytab` specifically on the node where the command is run. Perform validation of keytab file availability with desired name at the required location:

```
# stat /var/mmfs/tmp/krb5.keytab
File: /var/mmfs/tmp/krb5.keytab
Size: 502 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169898 Links: 1
Access: (0600/-rw-----) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 14:31:18.244837381 +0530
Modify: 2015-01-23 12:45:05.475837381 +0530
Change: 2015-01-23 12:45:05.476837381 +0530
Birth: -
```

3. Issue the `mmuserauth service create` command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-server-tls --enable-kerberos
--kerberos-server myKerberosServer --kerberos-realm example.com
```

The system displays the following output:

File authentication configuration completed successfully.

4. To verify the authentication configuration, issue the `mmuserauth service list` command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS                VALUES
-----
ENABLE_SERVER_TLS         true
ENABLE_KERBEROS           true
USER_NAME                  cn=manager,dc=example,dc=com
SERVERS                    myLDAPserver
NETBIOS_NAME               ess
BASE_DN                    dc=example,dc=com
USER_DN                    none
GROUP_DN                   none
NETGROUP_DN               none
```

```

USER_OBJECTCLASS      posixAccount
GROUP_OBJECTCLASS    posixGroup
USER_NAME_ATTRIB     cn
USER_ID_ATTRIB       uid
KERBEROS_SERVER      myKerberosServer
KERBEROS_REALM       example.com

```

```

OBJECT access not configured
PARAMETERS          VALUES
-----

```

5. Verify the user resolution on the system:

```

# id ldapuser3
uid=1002(ldapuser3) gid=1002(ldapuser3) groups=1002(ldapuser3)

```

Configuring LDAP without TLS and Kerberos for file access:

You can configure LDAP without TLS or Kerberos for file access. But this method is less secured compared to LDAP with TLS, LDAP with TLS and Kerberos, and LDAP with Kerberos configurations.

Provides an example on how to configure LDAP without TLS and Kerberos as the authentication method for file access.

1. Issue the **mmuserauth service create** command as shown in the following example:

```

# mmuserauth service create --type ldap --data-access-method file
--servers 192.0.2.18 --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret --netbios-name ess

```

The system displays the following output:

```
File Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```

FILE access configuration : LDAP
PARAMETERS          VALUES
-----
ENABLE_SERVER_TLS   false
ENABLE_KERBEROS     false
USER_NAME           cn=manager,dc=example,dc=com
SERVERS             192.0.2.18
NETBIOS_NAME        ess
BASE_DN             dc=example,dc=com
USER_DN             none
GROUP_DN            none
NETGROUP_DN        none
USER_OBJECTCLASS    posixAccount
GROUP_OBJECTCLASS   posixGroup
USER_NAME_ATTRIB    cn
USER_ID_ATTRIB      uid
KERBEROS_SERVER     none
KERBEROS_REALM     none

OBJECT access not configured
PARAMETERS          VALUES
-----

```

Configuring NIS-based authentication

The NIS-based authentication is useful in NFS only environment where NIS acts as an ID mapping server and also used for netgroups. When file access is configured with NIS, SMB access cannot be enabled.

Ensure that you have the following details before you start NIS-based authentication:

- NIS server name. This is case-specific
- IP address or host name of the NIS server
- Primary DNS is added in the `/etc/resolv.conf` file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, refer the corresponding Operating System documentation.

You need to run the **mmuserauth service create** command with the following mandatory parameters to configure NIS as the authentication method:

- `--type nis`
- `--data-access-method file`
- `--domain domainName`
- `--servers comma-delimited IP address or host name`

For more information on each parameter, see the **mmuserauth service create** command.

Provides an example on how to configure NIS as the authentication method for file access.

1. Issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type nis --data-access-method file
--servers myNISserver --domain nisdomain3
```

The system displays the following output:

```
File Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : NIS
PARAMETERS                VALUES
-----
SERVERS                    myNISserver
DOMAIN                     nisdomain3

OBJECT access not configured
PARAMETERS                VALUES
-----.
```

Managing user-defined authentication

In the user-defined mode of authentication, the user is free to select the authentication and ID mapping methods of their choice. It is the responsibility of the administrator of the client system to manage the authentication and ID mapping for file (NFS and SMB) and object access to the IBM Spectrum Scale system.

The IBM Spectrum Scale system administrators are not allowed use any of the GPFS commands to manage authentication. It is important for the end user to be aware of the limitations, if any, of the authentication and ID mapping scheme that will be implemented after configuring the user-defined mode of authentication.

The user-defined mode is appropriate in the following circumstances:

- The client already has protocol deployments either on GPFS installations or on different systems and is planning to move to using the protocol stack on the IBM Spectrum Scale system. The client wants to replicate the current authentication and ID mapping configuration. In this case, the client system administrator must be familiar with the required configuration settings that will be applied to the system.
- If the end user wants an authentication method that is not supported by the IBM Spectrum Scale system.

Note: If the end user wants to configure the authentication methods that are supported by the IBM Spectrum Scale system, it is highly recommended to configure the authentication and ID mapping methods by using the **mmuserauth** command instead of opting for the user-defined method of authentication.

The IBM Spectrum Scale system administrator needs to specify that the user-defined mode of authentication is used by using the **--type userdefined** option in the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type userdefined --data-access-method file
File Authentication configuration completed successfully.
```

Submit the **mmuserauth service list** command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
FILE access configuration : USERDEFINED
PARAMETERS                VALUES
-----
OBJECT access not configured
PARAMETERS                VALUES
-----
```

Typically, user-defined authentication is used when existing GPFS customers are already using GPFS with NFS and do not want to alter the authentication that is already configured on these systems. You can configure user-defined authentication for both object and file access or for object or file alone.

Note: Authorization depends upon authentication and ID mapping that is configured with the system. That is, the ACL control on exports, files, and directories depend on the authentication method that is configured.

Ensure the following while using the user-defined mode of authentication for file access:

- Ensure that the authentication server and ID mapping server are always reachable from all the protocol nodes. For example, if NIS is configured as the ID mapping server, you can use the 'ypwhich' command to ensure that NIS is configured and reachable from all the protocol nodes. Similarly, if LDAP is configured as authentication and ID mapping server, you can bind to the LDAP server from all protocol nodes to monitor if the LDAP server is reachable from all protocol nodes.
- Ensure that the implemented authentication and ID mapping configuration is always consistent across all the protocol nodes. This requires that the authentication server and ID mapping server are manually maintained and monitored by the administrator. The administrator must also ensure that the configuration files are not overwritten due to node restart and other similar events.
- Ensure that the implemented authentication and ID mapping-related daemons and processes across the protocol nodes are always up and running.
- The users or groups, accessing the IBM Spectrum Scale system over NFS and SMB protocols must resolve to a unique UID and GID respectively on all protocol nodes, especially in implementations where different servers are used for authentication and ID mapping. The name that is registered in ID mapping server for user and group must be checked for resolution.

For example:


```
# id fileuser
uid=1234(fileuser) gid=5678(filegroup) groups=5678(filegroup)
```

Note: However, there are some use cases where only NFSV3 based access to the IBM Spectrum Scale system is used. In such cases, the user and group IDs are obtained from the NFS client and there is no ID mapping setting is configured on the protocol nodes.

- If the IBM Spectrum Scale system is configured for multiprotocol support (that is, the same data is accessed through both NFS and SMB protocols), ensure that the IDs of users and groups are consistent across the NFS clients and SMB clients and that they resolve uniquely on the protocol nodes.
- Ensure that there is no conflict of UID and GID across users and groups that are accessing the system. This must be strictly enforced, especially in multiprotocol-based access deployments.
- Ensure that the Kerberos configuration files, placed on all protocol nodes, are in synchronization with each other. Ensure that the clients and the IBM Spectrum Scale system are part of the same Kerberos realm or trusted realm.
- While deploying two or more IBM Spectrum Scale clusters, ensure that the ID mapping is consistent in cases where you want to use IBM Spectrum Scale features like AFM, AFM-DR, and asynchronous replication of data.

The following table provides an overview of the authentication requirements for each file access protocol. Refer this table when you plan to use user-defined mode as the authentication method.

Table 7. Authentication requirements for each file access protocol.

File access protocol	Requirements
NFSV3	In scenarios where user name and group name are expected to be used to native GPFS commands (for example, setting data ownership, listing user or group quota), the IBM Spectrum Scale system must be able to resolve the UID and GID to user name and group name and vice versa, consistently across all the protocol nodes. Note: However, there are some use cases where only the NFSv3 based access to the IBM Spectrum Scale system is used. In such cases, the user and group IDs are coming from the NFS client and there is no ID mapping setting is configured on the protocol nodes.
Kerberos NFSV3	Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients. Ensure that the time is synchronized on the NFS server, NFS clients, and Kerberos server. Note: User names and group names are case-sensitive.
NFSV4	Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients. Domain name must be specified in the etc/idmapd.conf file and it must be the same on both the NFS server and NFS clients. Note: User names and group names are case-sensitive.

Table 7. Authentication requirements for each file access protocol. (continued)

File access protocol	Requirements
Kerberos NFS V4	<p>Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.</p> <p>Ensure that the time is synchronized on the NFS server, NFS clients, and Kerberos server.</p> <p>Domain name and local-realms must be specified in the <code>etc/idmapd.conf</code> file and it must be the same on both the NFS server and NFS clients.</p> <p>The value of "local-realms" takes the value of Kerberos realm with which the IBM Spectrum Scale system protocol nodes are configured.</p>
SMB	<p>Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.</p> <p>While integrating with non-windows server, ensure that the samba attributes are populated on the directory server for every user and group that are planning to access the IBM Spectrum Scale system. Special care must be taken to match the samba domain SIDs.</p> <p>For Kerberized SMB access, ensure that time is synchronized the SMB server, SMB client, and Kerberos server.</p>

The user-defined mode for object authentication integrates IBM Spectrum Scale Object Storage with the externally hosted keystone server. Ensure the following while using the user-defined mode of authentication for object access:

- Integration with external keystone server is supported over http only. No support for external keystone over SSL.
- The specified object user must be defined while enabling and configuring object in the external keystone server.
- The 'service' tenant must be defined in external keystone server.
- The 'admin' role must be defined in the external keystone server.
- Ensure that the specified swift user has 'admin' role in 'service' tenant.
- Object storage service endpoints must be correctly defined in the external keystone server.

For example, the external keystone server must contain the following endpoint for object-store:

```
# openstack endpoint list
```

ID	Region	Service Name	Service Type	Enabled	Interface	URL
c36e..9da5	None	keystone	identity	True	public	http://specscaleswift:5000/
f4d6..b040	None	keystone	identity	True	internal	http://specscaleswift:35357/
d390..0bf6	None	keystone	identity	True	admin	http://specscaleswift:35357/
2e63..f023	None	swift	object-store	True	public	http://specscaleswift:8080/v1/AUTH_%(tenant_id)s
cd37..9597	None	swift	object-store	True	internal	http://specscaleswift:8080/v1/AUTH_%(tenant_id)s
a349..58ef	None	swift	object-store	True	admin	http://specscaleswift:8080

If the object authentication is set to 'user-defined' and an IP address/port number is set in the proxy server configuration for keystone authentication, then that IP address will be checked using a simple http(s) request. If the request fails, then the AUTH_OBJ state will be set to "degraded" and an 'external keystone URL failure' event will be logged. This will not cause the node to be flagged as bad nor will it cause any public IP movement.

- Issue the following command:

```
mmces state show
```

The system displays output similar to this:

NODE	AUTH	AUTH_OBJ	NETWORK	NFS	OBJ	SMB	CES
spectrum-31.localnet.com	DISABLED	DEGRADED	HEALTHY	DISABLED	DEGRADED	HEALTHY	DEGRADED

- Issue the following command:

```
mmces events list
```

The system displays output similar to this:

NODE	TIMESTAMP	EVENT NAME	SEVERITY	DETAILS
spectrum-31.localnet.com	2015-10-18 18:23:05.386336--1:-1CEST	ks_url_exfail	WARNING	Keystone request failed using http://10.11.0.1:35357/v2.0

Configuring authentication for object access

You can use the following authentication methods for object access:

- Active Directory (AD)
- LDAP
- Local authentication

The AD-based and LDAP-based authentication methods use an external AD and LDAP server respectively to manage the authentication. Local authentication is handled by a Keystone server that resides within the IBM Spectrum Scale system.

The IBM Spectrum Scale system installation process configures Keystone server that is required for object access. If you are not using an external Keystone server, the IBM Spectrum Scale installation process by default configures the object authentication with local authentication as the authentication method.

Before you configure object authentication method, ensure that the Keystone Identity service is properly configured.

Note: Before you configure an authentication method for object access, ensure that all protocol nodes have CES IP addresses assigned and you are issuing the authentication configuration command from the protocol node for which a CES IP is already assigned.

Before you start manually configuring authentication method for object access, ensure that the `openldap-clients` RPM is installed.

The mapping between user, role, and tenant is stored in a database. This mapping is not deleted until you run the `mmuserauth service remove` command with the `--idmdelete` option for `data-access-method` object. While switching from one authentication type to another, it is mandatory to remove the mapping after removing the authentication.

Note:

It is recommended to run the `mmuserauth service check` command as follows after configuring object authentication using the `mmuserauth service create` command:

```
mmuserauth service check --data-access-method object -N cesNodes
```

If the `mmuserauth service check` command reports that any certificate file is missing on any of the nodes, then run the following command:

```
mmuserauth service check --data-access-method object -N cesNodes --rectify
```

For more information about `mmuserauth service check`, see the topic `mmuserauth command` in the *IBM Spectrum Scale: Administration and Programming Reference*.

Configuring local authentication for object access

Object access can be configured with the Keystone server that is available in the IBM Spectrum Scale system. In this mode, Keystone stores the identity and assignment information locally in its database.

The local authentication method is useful when you want to create and maintain a separate set of users for only object access. These users cannot use the local authentication credentials for accessing file data that is hosted through NFS and SMB protocols. If you want to allow a user to access both file and object, use an external authentication server such as AD or LDAP to manage user accounts and authentication requests.

Note: You cannot configure both file and object authentication method at one go, even if the authentication server is the same.

You need to use the **mmuserauth service create** command with the following mandatory parameters to configure local authentication for object access:

- `--type local`
- `--data-access-method object`
- `--ks-dns-name keystoneDNSName`
- `--ks-admin-user keystoneAdminName`
- `--ks-admin-pwd keystoneAdminPwd`. If not provided, the system prompts to enter the password during the command execution.
- `--enable-ks-ssl`, if SSL needs to be enabled.
- `--enable-ks-casigning`, if you want to use external CA signed certificate for token signing.

For more information on each parameter, see the **mmuserauth service create** command.

1. To configure local authentication for object access, issue **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --data-access-method object --type local
--ks-dns-name c40bbc2xn3 --ks-admin-user admin --ks-admin-pwd Passw0rd
```

The system displays the following output:

```
Object configuration with local (Database) as identity backend is completed
successfully.
Object Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS          VALUES
-----

OBJECT access configuration : LOCAL
PARAMETERS          VALUES
-----
ENABLE_KS_SSL       false
ENABLE_KS_CASIGNING false
KS_ADMIN_USER       admin
```

Configuring local authentication with SSL for object access:

Use the following steps to configure object access with the Keystone server that is available in the IBM Spectrum Scale system with SSL enabled.

1. Obtain certificates from the Certification Authority (CA) and place them at the following location on the current node from where the **mmuserauth service create** command is being executed.

```
/var/mmfs/tmp/ssl_cert.pem  
/var/mmfs/tmp/ssl_key.pem  
/var/mmfs/tmp/ssl_cacert.pem
```

Note:

- Self-signed certificates can be used for testing and demonstration purposes. However, the use of externally signed certificates is strongly recommended for production environments.
 - The name in the SSL certificate must match the Keystone endpoint name.
2. Remove existing local authentication for object access as follows.

```
mmuserauth service remove --data-access-method object
```
 3. Configure local authentication with SSL for object access as follows.

```
mmuserauth service create --data-access-method object --type local --enable-ks-ssl
```

Local authentication is now configured for object access with SSL enabled.

To disable SSL and configure local authentication for object access again, use the following steps.

4. Remove existing local authentication for object access as follows.

```
mmuserauth service remove --data-access-method object
```

If you are also changing authentication type, remove authentication and ID mappings by using the following commands in sequence.

```
mmuserauth service remove --data-access-method object  
mmuserauth service remove --data-access-method object --idmapdelete
```

5. Configure local authentication without SSL for object access as follows.

```
mmuserauth service create --data-access-method object --type local
```

Configuring an AD-based authentication for object access

You can configure Keystone with an external AD server as the authentication back-end so that AD users can access the object store by using their AD credentials. The same AD server can be used for both object access and file access.

The AD server is set up to handle the authentication requests. AD is used as an LDAP server. Unlike file access, multiple AD domains are not supported.

Prerequisites

Ensure that you have the following details before you start AD-based authentication configuration:

- AD server details such as IP address or host name, user name, user password, base dn, and user dn.
- If you want to configure TLS with AD for secure communication between Keystone and AD, you need to place the CA certificate that is used for signing the AD server setup for TLS under the following directory of the node on which the **mmuserauth service create** command is run:
 - /var/mmfs/tmp/ldap_cacert.pem
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.

See “Integrating with AD server” on page 70 for more information on the prerequisites for integrating AD server with the IBM Spectrum Scale system.

The following are the mandatory parameters to be used with **mmuserauth service create** command to configure AD-based authentication for object access:

- **--type ad**

- **--data-access-method** object
- **--servers** IP address or host name of AD. All user lookups by Keystone are done only against this server. If multiple servers are specified, only the first server is used and the rest are ignored.
- **--base-dn** ldapBase
- { **--enable-anonymous-bind** | **--user-name BindDN --password BindPwd**} (You need to mention either anonymous bind or either `--user-name` or `--password`)
- **--enable-server-tls**, if TLS needs to be enabled
- **--user-dn** ldapUserSuffix (LDAP container from where users are looked up)
- **--ks-dns-name** keystoneDNSName
- **--ks-admin-user** keystoneAdminUser from AD
- **--enable-ks-ssl**, if SSL needs to be enabled. You need to have another set of certificates that are placed in standard directory.
- **--enable-ks-casigning**, if you want to use external CA signed certificate for token signing
- **--ks-swift-user** Swift_Service_User from AD
- **--ks-swift-pwd** Swift_Service_User's Password from AD

For more information on each parameter, see the **mmuserauth service create** command.

To change the authentication method that is already configured for object access, you need to remove the authentication method and ID mappings. For more information, see “Deleting the authentication and the ID mapping configuration” on page 105.

Configuring AD without TLS for object access:

Configuring AD without TLS does not provide secured communication between the IBM Spectrum Scale system and the authentication server.

1. Submit the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
--base-dn "dc=IBM,DC=local" --ks-dns-name c40bbc2xn3 --ks-admin-user admin
--servers myADserver --user-id-attrib cn --user-name-attrib sAMAccountName
--user-objectclass organizationalPerson --user-dn "cn=Users,dc=IBM,dc=local"
--ks-swift-user swift --ks-swift-pwd Passw0rd
```

The system displays the following output:

```
Object configuration with LDAP (Active Directory) as identity
backend is completed successfully.
Object Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration: AD
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS     false
ENABLE_KS_SSL         false
USER_NAME             cn=Administrator,cn=Users,dc=IBM,dc=local
SERVERS               myADserver
```

```

BASE_DN          dc=IBM,DC=local
USER_DN          cn=users,dc=ibm,dc=local
USER_OBJECTCLASS organizationalPerson
USER_NAME_ATTRIB sAMAccountName
USER_ID_ATTRIB   cn
USER_MAIL_ATTRIB mail
USER_FILTER      none
ENABLE_KS_CASIGNING false
KS_ADMIN_USER    admin

```

Configuring AD with TLS for object access:

Configuring TLS with AD helps to encrypt the communication between the IBM Spectrum Scale system and AD server.

Configures AD with TLS as the authentication method for object access.

1. Ensure that the CA certificate for AD server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem` specifically on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```

# stat /var/mmfs/tmp/ldap_cacert.pem
File: /var/mmfs/tmp/ldap_cacert.pem
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530

```

2. To configure AD with TLS authentication for object access, issue the `mmuserauth service create` command as shown in the following example:

```

# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
--base-dn "dc=IBM,DC=local" --enable-server-tls --ks-dns-name c40bbc2xn3
--ks-admin-user admin --servers myADserver --user-id-attrib cn
--user-name-attrib sAMAccountName --user-objectclass organizationalPerson
--user-dn "cn=Users,dc=IBM,dc=local" --ks-swift-user swift
--ks-swift-pwd PasswOrd

```

The system displays the following output:

```

Object configuration with LDAP (Active Directory) as identity
backend is completed successfully.
Object Authentication configuration completed successfully.

```

3. To verify the authentication configuration, issue the `mmuserauth service list` command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```

FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration: AD
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS     true
ENABLE_KS_SSL         false
USER_NAME             cn=Administrator,cn=Users,dc=IBM,dc=local
SERVERS              myADserver
BASE_DN              dc=IBM,DC=local
USER_DN              cn=users,dc=ibm,dc=local

```

USER_OBJECTCLASS	organizationalPerson
USER_NAME_ATTRIB	sAMAccountName
USER_ID_ATTRIB	cn
USER_MAIL_ATTRIB	mail
USER_FILTER	none
ENABLE_KS_CASIGNING	false
KS_ADMIN_USER	admin

Configuring an LDAP-based authentication for object access

You can configure Keystone server with an external LDAP server as the authentication server so that users in LDAP can access object store with their LDAP credentials. This configuration is useful when the user credentials are stored in LDAP and you want them to get authenticated to access the object store. It is also useful in cases where you have configured file authentication with LDAP and want to use the same authentication for object access. You can use TLS with the LDAP server to further tighten the security of the system.

Prerequisites

Ensure that you have the following details before you configure LDAP-based authentication:

- LDAP server details such as IP address or host name, LDAP user name, user password, base dn, and user dn.
- If you want to configure TLS with LDAP for secure communication between Keystone and LDAP, you need to place the CA certificate that is used for signing the LDAP server setup for TLS under the following directory of the node on which the **mmuserauth service create** command is run:
 - /var/mmfs/tmp/ldap_cacert.pem
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.

See “Integrating with LDAP server” on page 71 for more information on the prerequisites for integrating LDAP server with the IBM Spectrum Scale system.

You need to issue the **mmuserauth service create** command to configure LDAP-based authentication with the following mandatory parameters:

- **--type** ldap
- **--data-access-method** object
- **--servers** IP address or host name of LDAP (all user lookups by Keystone is done only against this server. If multiple servers are specified, only the first server is used and rest are ignored).
- **--base-dn** ldapBase
- { **--enable-anonymous-bind** | **--user-name BindDN --password BindPwd** } (You need to mention either anonymous bind or either **--user-name** or **--password**).
- **--enable-server-tls**, if TLS needs to be enabled.
- **--user-dn** ldapUserSuffix (LDAP container from where users are looked up)
- **--ks-dns-name** keystoneDNSName
- **--ks-admin-user** keystoneAdminUser from LDAP.
- **--enable-ks-ssl**, if SSL needs to be enabled. You need to have another set of certificates that are placed in the standard directory.
- **--enable-ks-casigning**, if you want to use external CA signed certificate for token signing.
- **--ks-swift-user** swiftServiceUser from LDAP.
- **--ks-swift-pwd** swiftServiceUser Password from LDAP.

For more information on each parameter, see the **mmuserauth service create** command.

To change the authentication method that is already configured for object access, you need to remove the authentication method and ID mappings. For more information, see “Deleting the authentication and the ID mapping configuration” on page 105.

Configuring LDAP without TLS for object access:

Perform the following steps to configure LDAP-based authentication for object access:

1. To configure LDAP-based authentication for object access, issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=essldapdomain" --password "Passw0rd"
--base-dn dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com --ks-dns-name c40bbc2xn3
--ks-admin-user mamdouh --servers 192.0.2.11
--user-dn "ou=People,dc=essldapdomain" --ks-swift-user swift
--ks-swift-pwd Passw0rd
```

The system displays the following output:

```
Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration : LDAP
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS     false
ENABLE_KS_SSL         false
USER_NAME             cn=manager,dc=essldapdomain
SERVERS               192.0.2.11
BASE_DN               dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com
USER_DN               ou=people,dc=essldapdomain
USER_OBJECTCLASS      posixAccount
USER_NAME_ATTRIB      cn
USER_ID_ATTRIB        uid
USER_MAIL_ATTRIB      mail
USER_FILTER           none
ENABLE_KS_CASIGNING   false
KS_ADMIN_USER         mamdouh
```

Configuring LDAP with TLS for object access:

Perform the following steps to configure LDAP with TLS-based authentication for object access:

1. Ensure that the CA certificate for the LDAP server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem` specifically on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```
# stat /var/mmfs/tmp/ldap_cacert.pem
File: /var/mmfs/tmp/ldap_cacert.pem
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
```

```
Context: unconfined u:object r:user tmp t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. To configure LDAP with TLS-based authentication for object access, issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=essldapdomain" --password "Passw0rd"
--base-dn dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com --enable-server-tls
--ks-dns-name c40bbc2xn3 --ks-admin-user mamdouh --servers 192.0.2.11
--user-dn "ou=People,dc=essldapdomain" --ks-swift-user swift
--ks-swift-pwd Passw0rd
```

The system displays the following output:

```
Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.
```

3. To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration : LDAP
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS     true
ENABLE_KS_SSL         false
USER_NAME             cn=manager,dc=essldapdomain
SERVERS               192.0.2.11
BASE_DN               dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com
USER_DN               ou=people,dc=essldapdomain
USER_OBJECTCLASS      posixAccount
USER_NAME_ATTRIB     cn
USER_ID_ATTRIB        uid
USER_MAIL_ATTRIB      mail
USER_FILTER           none
ENABLE_KS_CASIGNING   false
KS_ADMIN_USER         mamdouh
```

Configuring object authentication with an external Keystone server

Object can be configured with an external Keystone server. This is done when either there is already an existing Keystone server that is deployed in the environment or when the system administrator wants to host a keystone server externally so that it can be used for other OpenStack services.

The following prerequisites must be met before you start configuring an external Keystone server with the IBM Spectrum Scale system.

- The external Keystone Server must be running and reachable from all protocol nodes.
- The Keystone server administrator must create an Object Storage service for the required user, for Object authentication configuration.

To configure an external Keystone server with the IBM Spectrum Scale system, issue the **mmuserauth service create** command as shown in the following example:

```
mmuserauth service create --data-access-method object --type
userdefined --ks-swift-user <SWIFTserviceUser> --ks-swift-pwd <SWIFTserviceUserpassword>
--ks-ext-endpoint <endpoint of keystone server>
```

Configuring an external Keystone server for object authentication when using the installation toolkit

If you plan to configure authentication for IBM Spectrum Scale for object storage with an external Keystone server and you are using the installation toolkit, do the following steps:

1. When configuring IBM Spectrum Scale for object storage with the installation toolkit, do not configure object authentication with external Keystone.

After successful installation and deployment, IBM Spectrum Scale for object storage is configured with local authentication.

2. Run the following commands to configure object authentication with external Keystone:

```
mmuserauth service remove --data-access-method object

mmuserauth service remove --data-access-method object --idmapdelete

mmuserauth service create --data-access-method object --type userdefined
--ks-ext-endpoint http://specscaleswift:35357/v3
--ks-swift-user swift --ks-swift-pwd password
```

Note: Cleaning up authentication leads to loss of data access to the end clients. For example, in the preceding command sequence, client access to data created with local authentication enabled is lost when you remove local authentication before configuring external Keystone.

Managing object users, roles, and projects

The Object Storage service of the IBM Spectrum Scale system uses the Keystone service for identity management. The identity management service consists of user authentication and authorization processes.

You can use an external Microsoft Active Directory or LDAP server or a local database as the back-end to store and manage user credentials for user authentication. The authorization details such as relation of users with projects and roles are maintained locally by the keystone server. The customer can select the authentication server to be used. For example, if AD is already existing in an enterprise deployment and the users in AD are required to access object data, the customer opts for AD as the back-end authentication server.

When the back-end authentication server is AD or LDAP, the user management operations such as creating or deleting a user are the responsibility of the AD/LDAP administrator, who can optionally also be the Keystone server administrator. When local authentication is used for object access, the user management operations are done by the Keystone administrator. In case of authorization, the management tasks such as creating roles, projects, and associating the user with them is done by the Keystone Administrator. The Keystone administration can be done through the Keystone V3 REST API or by using an OpenStack python-based client.

Before you start creating object users, and projects, ensure that Keystone server is configured and the authentication servers are set up properly.

Note:

- If the cluster is reachable from the system, the OpenStack command can be issued from any system.
- If the OpenStack command is run from any of the protocol nodes, then you can use the `openrc` file to set the required environment that is used by OpenStack commands to manage the Keystone server. The advantage of using the `openrc` file is that you are not required to enter the following details every time you enter the commands: `--os-identity-api-version`, `--os-username`, `--os-password`, `--os-project-domain-name`, `--os-user-domain-name`, `--os-domain-id default`, and `--os-auth-url`.
- The user create, update, and delete operations are only applicable when local authentication method is used for object access.

- For more information on the Keystone V3 REST API, see the OpenStack API Documentation (developer.openstack.org/api-ref-identity-v3.html).

Creating a new user

Use the **openstack user create** command as shown in the following example to create new user in the local database to support local authentication for object access:

```
# openstack --os-identity-api-version 3 --os-username admin --os-password
Passw0rd --os-project-domain-name Default --os-user-domain-name Default --osdomain-
id default --os-auth-url http://specscaleswift:35357/v3 user create --password-prompt --
email newuser1@localdomain.com --domain default newuser1
User Password:
Repeat User Password:
```

Field	Value
domain_id	default
email	newuser1@localdomain.com
enabled	True
id	2a3ef8031359457292274bcd70e34d00
links	{u'self': u'http://specscaleswift:35357/v3/users/2a3ef8031359457292274bcd70e34d00'}
name	newuser1

GUI navigation

To work with this function in the IBM Spectrum Scale GUI, log on to the GUI and select **Object > Users**.

Listing users

Use the **openstack user list** command as shown in the following example to list users who are created in the local database:

```
# source $HOME/openrc
# openstack user list
```

ID	Name
2a3ef8031359457292274bcd70e34d00	newuser1
a95783144edd414aa236a3d1582a3067	admin

Changing the password of a user

Use the **openstack user set** command to update the object user details. The following example shows how to change the password:

```
# openstack user set --password Passw0rd newuser2
```

Deleting a user

Use the **openstack user delete** command as shown in the following example to delete the users who are created in the local database:

```
# openstack user delete newuser2
```

Listing user roles

Use the **openstack role list** command as shown in the following example to list the user roles:

```
# openstack role list
+-----+-----+
| ID | Name |
+-----+-----+
| ed38022b46094a51918e6e46f87e7290 | admin |
+-----+-----+
```

Creating a new role

Perform the following steps to create a new user role:

1. Issue the **openstack role create** command to create a new user role:

```
#openstack role create member
+-----+-----+
| Field | Value |
+-----+-----+
| id | 1f14f95826fe4c8590760b3d3e4ce7e0 |
| links | {'self': u'http://specscaleswift:35357/v3/roles/1f14f95826fe4c8590760b3d3e4ce7e0'} |
| name | member |
+-----+-----+
```

2. Verify the newly created role by using the **openstack role list** command:

```
# openstack role list
+-----+-----+
| ID | Name |
+-----+-----+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member |
| ed38022b46094a51918e6e46f87e7290 | admin |
+-----+-----+
```

GUI navigation

To work with this function in the IBM Spectrum Scale GUI, log on to the GUI and select **Object > Roles**.

Assigning a role to a user

Perform the following steps to assign a user role to a user:

1. Issue the **openstack role add** command to assign role to a user as shown in the following example:

```
# openstack role add --user newuser1
--domain default member
```

2. Submit the **openstack role list** command to verify the user role of the user as shown in the following example:

```
# openstack role list --user newuser1
+-----+-----+
| ID | Name |
+-----+-----+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member |
+-----+-----+
```

Creating a new project, adding a user, and assigning a role to the user

Perform the following steps to create a new project and add a user to the project with a specified role:

1. Submit the **openstack project create** command to create a new project:

```
# openstack project create newproject
+-----+-----+
| Field | Value |
+-----+-----+
| description | |
| domain_id | default |
| enabled | True |
| id | 2dfcddb70b75435fb2015c86d46ffc0b |
+-----+-----+
```

```

| links          | {u'self':
u'http://specscaleswift:35357/v3/projects/2dfcddb70b75435fb2015c86d46ffc0b'} |
| name          | newproject
| parent_id     | None
+-----+-----+

```

2. Submit the **openstack role add** command to add a role to the user as shown in the following example:

```

# openstack role add --user newuser1 --
project newproject member

# openstack role add --user newuser1
--project newproject admin

```

3. Submit the **openstack role list** command to list the user roles as shown in the following example:

```

# openstack role list --user newuser1 --
project newproject
+-----+-----+-----+-----+
| ID                               | Name | Project | User |
+-----+-----+-----+-----+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member | newproject | newuser1 |
| ed38022b46094a51918e6e46f87e7290 | admin  | newproject | newuser1 |
+-----+-----+-----+-----+

```

Listing endpoints

Use the **openstack endpoint list** command as shown in the following example to view the endpoints that are available:

```

# openstack endpoint list
+-----+-----+-----+-----+-----+-----+
| ID          | Region | Service Name | Service Type | Enabled | Interface | URL |
+-----+-----+-----+-----+-----+-----+
| c36e..9da5 | None   | keystone     | identity     | True   | public    | http://specscaleswift:5000/
| f4d6..b040 | None   | keystone     | identity     | True   | internal  | http://specscaleswift:35357/
| d390..0bf6 | None   | keystone     | identity     | True   | admin     | http://specscaleswift:35357/
| 2e63..f023 | None   | swift        | object-store | True   | public    | http://specscaleswift:8080/v1/AUTH_%(tenant_id)s
| cd37..9597 | None   | swift        | object-store | True   | internal  | http://specscaleswift:8080/v1/AUTH_%(tenant_id)s
| a349..58ef | None   | swift        | object-store | True   | admin     | http://specscaleswift:8080
+-----+-----+-----+-----+-----+-----+

```

Creating object accounts

An account is used to group or isolate object resources. Each object user is part of an account. Object users are mapped to an account and it can access only the objects that reside within the project. Each user needs to be defined with a set of user rights and privileges to perform a specific set of operations on the resources of the account to which it belongs. Users can be part of multiple accounts, but it is mandatory that a user must be associated with at least one account.

You must create at least one account before adding users. An account contains a list of containers in the object storage. You can also define quota at the account level. An object account represents a storage location for a project rather than a specific user.

GUI navigation

To work with this function in the IBM Spectrum Scale GUI, log on to the GUI and select **Object > Accounts**.

1. To view the details for an existing account, issue the **swift stat** command:

```

# swift stat --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3 \
--os-project-name admin \
--os-project-domain-name Default \
--os-username admin \
--os-user-domain-name Default \
--os-password Passw0rd \
--auth-version 3

```

The system displays output similar to the following:

Account: AUTH_bea5a0c632e54eaf85e9150a16c443cet

2. To create a new account, do the following steps:
 - a. Use the **openstack project create** command to create a project.

For example, create the project 'salesproject' in the Default domain using the command:

```
# openstack project create salesproject --domain Default
```

The system displays output similar to the following:

Field	Value
description	
domain_id	default
enabled	True
id	ec4a0bff137b4c1fb67c6fe8fbb6a37b
name	salesproject

- b. Use the **openstack role add** command to associate roles to the users who need access to the project:

```
# openstack role add --user admin --project salesproject admin
```

The system displays output similar to the following:

Field	Value
description	
domain_id	default
enabled	True
id	ec4a0bff137b4c1fb67c6fe8fbb6a37b
name	salesproject

3. To see the new account details, issue the **swift stat** command with the new project value:

```
# swift stat --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3 \  
--os-project-name salesproject \  
--os-project-domain-name Default \  
--os-username admin \  
--os-user-domain-name Default \  
--os-password Passw0rd \  
--auth-version 3
```

The system displays output similar to the following:

Account: AUTH_ec4a0bff137b4c1fb67c6fe8fbb6a37b

Deleting expired tokens

By default, the Keystone Identity Service stores expired tokens in the database indefinitely. While potentially useful for auditing in production environments, the accumulation of expired tokens considerably increases the database size and might affect the service performance.

Use cron as follows to configure a periodic task on one of the protocol nodes that purges expired tokens hourly or based on the load in your environment.

```
# (crontab -l -u keystone 2>&1 | grep -q token_flush) || \  
echo '@hourly /usr/bin/keystone-manage token_flush >/var/log/keystone/keystone-tokenflush.log 2>&1' \  
>> /var/spool/cron/keystone
```

Deleting the authentication and the ID mapping configuration

Deleting the authentication and ID mapping configuration results in loss of access to data. Before you remove or edit ID mappings, determine how access to data is going to be maintained.

Note: You are not allowed to delete both the authentication configuration and the ID mappings at the same time. You need to remove the authentication configuration first and then the ID maps. The system does not allow you to delete the ID maps without deleting the authentication configuration.

The following example shows how to remove the authentication configuration.

1. Issue the **mmuserauth service list** command to see the authentication method that is configured in the system:

```
# mmuserauth service list
FILE access configuration: LDAP
PARAMETERS VALUES
-----
ENABLE_ANONYMOUS_BIND false
ENABLE_SERVER_TLS false
ENABLE_KERBEROS false
USER_NAME cn=manager,dc=example,dc=com
SERVERS 10.0.100.121
NETBIOS_NAME eslnode
BASE_DN dc=example,dc=com
USER_DN ou=people,dc=example,dc=com
GROUP_DN none
NETGROUP_DN ou=netgroup,dc=example,dc=com
USER_OBJECTCLASS inetOrgPerson
GROUP_OBJECTCLASS posixGroup
USER_NAME_ATTRIB cn
USER_ID_ATTRIB uid
KERBEROS_SERVER none
KERBEROS_REALM none
OBJECT access not configured
PARAMETERS VALUES
-----
```

2. Issue the **mmuserauth service remove** command to remove the authentication configuration as shown in the following example:

```
# mmuserauth service remove --data-access-method file
mmcesuserauth service remove: Command successfully completed.
```

3. Issue the **mmuserauth service list** command to verify whether the authentication configuration is removed:

```
# mmuserauth service list
FILE access not configured
PARAMETERS VALUES
-----
OBJECT access not configured
PARAMETERS VALUES
-----
```

For more information, see *For more information, see `mmuserauth command` in the IBM Spectrum Scale: Administration and Programming Reference*

Deleting authentication configuration as shown in the previous example does not delete the ID maps. Use the **--idmapdelete** option with the **mmuserauth service remove** command to remove ID maps that are created for user authentication:

```
# mmuserauth service remove --data-access-method file --idmapdelete
mmuserauth service remove: Command successfully completed
```

The deletion of ID maps that are used for file access is only applicable when AD with Automatic ID mapping or RFC2307 ID mapping.

Deleting ID maps might also be required in the case of object access. ID map delete option can be used if the system administrator wants to clean up the entire Keystone authentication configuration, including the mapping of users with projects and roles. Cleaning up of ID mapping information results in loss of access to any existing data that is being accessed through the Object Storage interface. Deleting ID

mappings deletes user-role-projects mappings as well. Without these mappings, new users are unable to access the old data unless the keystone administrator creates the mapping again for the new user. ID maps are deleted in environments where the object protocol needs to be removed or the entire object store needs to be erased. This is usually done in preproduction or test environments.

If you want to change the authentication method that is already configured for object access, you must remove the authentication method and ID mappings by issuing the **mmuserauth service remove --data-access-method object** and **mmuserauth service remove --data-access-method object --idmapdelete** commands in sequence, as shown in the following example:

```
# mmuserauth service remove --data-access-method object
mmuserauth service remove: Command successfully completed

# mmuserauth service remove --data-access-method object --idmapdelete
mmuserauth service remove: Command successfully completed

# mmuserauth service list
FILE access not configured
PARAMETERS VALUES
-----
OBJECT access not configured
PARAMETERS VALUES
-----
```

Note: When you delete the ID maps that are created for file or object access, ensure that all the protocol nodes are in the healthy state. You can view the health status of protocol nodes by using the **mmces state show -a** command.

Listing the authentication configuration

Use the **mmuserauth service list** command to see the authentication method that is configured in the system.

```
# mmuserauth service list
FILE access configuration : LDAP
PARAMETERS VALUES
-----
ENABLE_SERVER_TLS false
ENABLE_KERBEROS false
USER_NAME cn=manager,dc=example,dc=com
SERVERS 9.122.123.172
NETBIOS_NAME eslnode
BASE_DN dc=example,dc=com
USER_DN ou=people,dc=example,dc=com
GROUP_DN none
NETGROUP_DN ou=netgroup,dc=example,dc=com
USER_OBJECTCLASS inetOrgPerson
GROUP_OBJECTCLASS posixGroup
USER_NAME_ATTRIB cn
USER_ID_ATTRIB uid
KERBEROS_SERVER none
KERBEROS_REALM none
OBJECT access not configured
PARAMETERS VALUES
-----
```

For more information, see the topic *mmuserauth command* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Verifying the authentication services configured in the system

Use the **mmuserauth service check** command to check whether the authentication configuration is consistent across the cluster and the required services are enabled and running. This command validates and corrects the authentication configuration files and starts any associated services if needed.

You can check the following authentication details by using the **mmuserauth service check** command:

- Authentication method. Use the `--data-access-method {file|object|all}` option to check the authentication method.
- Authentication configuration on each node. Use the `[-N|--nodes] {node-list|cesNodes}` option to check this. If the specified node is not protocol node, the check operation gets ignored on that node. If protocol node is specified, then the system checks configuration on all protocol nodes. If you do not specify a node, the system checks the configuration of only the current node.
- Whether the authentication backend server is reachable. Use the `--server-reachability` option to verify this. If object is configured with external Keystone server, this check is not performed.
- Rectify the configuration if required by using the `[-r | --rectify]` option. Using this option rectifies the configuration for the specified nodes by copying any missing configuration files or SSL/TLS certificates from another node.

See the **mmuserauth service check** command for more information.

Example - File authentication check

Issue the **mmuserauth service check** command.

```
# mmuserauth service check --type file --nodes dgnode3,dgnode2
--server-reachability -r
```

```
Userauth file check on node: dgnode3
Checking SSSD_CONF: OK
LDAP server status
LDAP server 9.118.46.17 : OK
Service 'sssd' status: OK
Userauth file check on node: dgnode2
dgnode2: not CES node. Ignoring...
```

You can use the **id** command to see the list of users and groups fetched from the LDAP server. For example:

```
# id ldapuser2
uid=1001(ldapuser2) gid=1001(ldapuser2) groups=1001(ldapuser2)
```

Example - Object authentication check

Issue the **mmuserauth service check** command.

```
# mmuserauth service check
--server-reachability --data-access-method
object
Userauth object check on node: dgnode3
Checking keystone.conf: OK
LDAP servers status
LDAP server sonash1 : OK
Service 'keystone-all' status: OK
```

Modifying the authentication method

If data already exists or is created with the existing authentication and ID mapping method, it is not recommended to change the authentication or the ID mapping modes. Changing the authentication method also might invalidate the existing ACLs that are applicable to files and directories. ACLs depend on preexisting users and groups ID.

To modify the authentication method, perform the following steps:

1. List the existing authentication configuration for file and object authentication method by using the **mmuserauth service list** command.

2. Identify the parameters that you need to change. If an authentication method and ID maps are already existing, you must not plan to change the authentication type or ID mapping schemes. When you remove the existing authentication method and ID maps, the user and group of users who were accessing the data cannot access the data anymore.

The following list provides the parameters that can be modified in each authentication configuration.

For file authentication:

- With LDAP authentication, all attributes of the configuration can be modified. When changing authentication servers, ensure that the newly mentioned servers are the replica of the original servers, otherwise, it might result in loss of access to data.
- With AD authentication, all the attributes of the configuration can be modified. When changing the authentication server, ensure that the newly mentioned server is a domain controller in the same AD domain that is being served by the original server, otherwise, it might result in loss of access to data. If UNIX ID maps are specified in current configuration and more new AD domains are to be added, it is vital to specify the current list of domains along with the new domains.
- With NIS authentication, all attributes of the configuration can be modified. When changing servers, ensure that the newly mentioned servers are serving the same NIS domain as the original servers; otherwise, it might result in loss of access to data.

For object authentication:

You can change all options except **--data-access-method** and **--type** parameters.

3. Clean up the existing authentication by using the **mmuserauth service remove** command. Do not specify the **--idmapdelete** option as it results in loss of access to data.
4. Issue the **mmuserauth service create** with the required parameter change; ensuring that you use the same authentication, ID mapping scheme, and associated authentication servers.
5. List the authentication configuration by using the **mmuserauth service list** to verify the change.
6. Ensure that the authentication is consistent across the cluster by using the **mmuserauth service check** command.

Authentication limitations

Consider the following authentication limitations when you configure and manage the IBM Spectrum Scale system:

Object access limitations

For AD-based authentication for object access:

- Only single AD server is used. If the configured AD server is down, the Keystone authentication fails.
- Does not support multiple AD Domains.
- Only Windows 2008 R2 and later are supported.
- Only read access to the AD server is supported. That is, you are not authorized to create a new user and modify or delete an existing user from the IBM Spectrum Scale system. Only the AD server administrator can do these tasks.

For LDAP-based authentication for object access:

- Only single LDAP server is used. If the configured LDAP server is down, the Keystone authentication fails.
- Only LDAP servers compatible with LDAP RFC 4511 are supported.
- Only read access to the LDAP server is supported. That is, you are not authorized to create a new user and modify or delete an existing user from the IBM Spectrum Scale system. Only the LDAP server administrator can do these tasks.

File access limitations

AD based authentication

For AD with automatic ID mapping:

- No support for migrating the internally generated user and group ID maps to external ID mapping server. If data is stored on the IBM Spectrum Scale system with AD and automatic ID mapping, adding RFC2307 later requires the UIDs and GIDs that are used internally by the IBM Spectrum Scale system match the UIDs and GIDs stored in RFC2307. This is not possible if conflicting UIDs and GIDs are already stored in RFC2307. To avoid potential conflicts, configure the IBM Spectrum Scale system by using AD and RFC2307 right from the beginning.
- Although AD along with automatic ID mapping can be used to have same ID maps between systems that are in AFM relationship, this configuration does not serve as a complete replacement of RFC2307. This configuration can be used in a predominantly SMB only setup, where NFS users are not already present in the environment. If NFS users are preexisting in the customer environment and these users intend to access the data with SMB users, then RFC2307 is mandatory.
- When AD-based authentication is used, SMB protocol access is kerberized by default. Access the system by using the netbios name that is specified in the command.
- Kerberized NFSv3-based access is not supported with AD as an authentication server.

For AD with RFC2307:

- Enabling RFC2307 for a trusted domain requires a two-way trust between the native and the trusted domains.
- To access the IBM Spectrum Scale system, users and groups must have a valid UID/GID assigned to them in AD. For user access, the windows group membership are evaluated on the IBM Spectrum Scale system. Hence, accessing a user's primary group is considered as the Microsoft Windows Primary group and not the UNIX primary group that is listed in the UNIX attribute tab in the user's properties. Therefore, the user's primary Microsoft Windows group must be assigned with a valid GID.
- The **mmuserauth service create** command does not check the two-way trust between the native domain and the RFC2307 domain that is required for ID mapping services to function properly. The customer is responsible for configuring the two-way trust relationship between these domains. The customer is responsible for assigning UIDs to users and GIDs to groups. The command does not return an error if a UID or GID is not assigned.
- Multiprotocol access of protocol exports is only allowed between NFSV4 and SMB. That is, you cannot access the same export by using both NFSV3 and SMB protocols.
- Kerberized NFSv3-based access is not supported by AD as an authentication server.

LDAP based authentication

- Users with the same user name from different organizational units under the specified baseDN in the LDAP server are denied access to SMB shares irrespective of the LDAP user suffix and LDAP group suffix values configured on the system.
- If multiple LDAP servers are specified during configuration, at any point in time, only one LDAP server is used.
- LDAP referrals are not supported.
- ACL management through windows clients is not supported.
- Only LDAP servers that implement RFC2307 schema are supported.

General limitations for file access

- Whenever Samba service is stopped when AD with RFC2307 authentication is used on a protocol node, the NFS based access also get affected on that particular node.
- When using Microsoft Active Directory (AD) as an authentication system, the IBM Spectrum Scale system supports only the NetBIOS logon name for authentication and not the User Principle Name

(UPN). Active Directory replaces some of the special characters that are used in the UPN with the underscore character (hexadecimal value 0x5F) for the related NetBIOS logon name of the user. For the complete list of the special characters that are replaced in the NetBIOS logon name, see Microsoft Active Directory documentation. Follow these steps to locate the NetBIOS logon name for an Active Directory domain user:

1. From the Windows Start menu, select Administrative Tools > Active Directory Users and Computers
2. Right-click the Active Directory Domain user for which you require the NetBIOS logon name
3. Select Properties > Account Tab and check the value of the User logon name (pre-Windows 2000): field

- Authentication configuration commands restart the IBM Spectrum Scale protocol services such as SMB and NFS. The protocol services resume a few seconds after an authentication configuration command completes.
- For the file data access method, switching or migrating from one authentication method to another is not supported because it might lead to loss of access to the data on the system.
- The IBM Spectrum Scale system does not support authentication servers (AD, LDAP, and NIS) that are running on virtual machines that are stored on an NFS or SMB export. The IBM Spectrum Scale system requires the authentication server to be running, while configuring authentication and while serving connection requests over protocols. The virtualizer cannot boot the authentication server unless the protocols are configured for authentication and data is ready to be served over the exports.
- The length of a user name or a group name of the users and group of users who need to access the data cannot be more than 32 characters
- The NFSV4 clients must be configured with the same authentication and ID mapping server as that of the IBM Spectrum Scale system. The IBM Spectrum Scale system does not support an NFSV4 client configured with different authentication and ID mapping servers.
- In order to use NFSV4 ID mapping, the NFS ID map domain needs to be set on the IBM Spectrum Scale protocol nodes and the same NFS ID map domain must be configured on every NFS client. Below is an example of how to configure NFSV4 ID mapping.

1. Issue the **mmnfs configuration list** command.

The system will display this output showing that the ID map domain is not set:

```
idmap DOMAIN
      idmap Configuration
      -----
      DOMAIN:
```

2. Issue the **mmnfs configuration change DOMAIN=MY_IDMAP_DOMAIN** command to set the NFS ID map domain.
3. Issue the **mmnfs configuration list** command to verify that the ID map domain is set.

The system displays this output:

```
DOMAIN: MY_IDMAP_DOMAIN
```

Note: After running the **mmnfs configuration list** command, you will see another DOMAIN labeled DOMAINNAME. That DOMAINNAME is the domain for your authentication server (LDAP or AD). This domain name gets set when you use the IBM Spectrum Scale installer to set your authentication method. Below is an example of the system output.

```
NFS Ganesha Configuration:
-----
NFS_PROTOCOLS: 3,4
NFS_PORT: 2049
MNT_PORT: 20048
NLM_PORT: 0
RQUOTA_PORT: 0
LEASE_LIFETIME: 60
DOMAINNAME: ESSLDAPDOMAIN
DELEGATIONS: DISABLED
```

General authentication considerations

Consider the following NFSV4 authentication requirements when you configure and manage the IBM Spectrum Scale

NSFv4 considerations

For NSFv4 authentication for file access:

- The NFSV4 clients must be configured with the same authentication and ID mapping server as that of the IBM Spectrum Scale system. The IBM Spectrum Scale system does not support an NFSV4 client configured with different authentication and ID mapping servers.

Configuring with the spectrumscale installation toolkit

You can use the configuration options of the **spectrumscale** installation toolkit to configure GPFS and protocols on an ongoing basis, as an alternative to the other GPFS cluster creation and configuration commands.

For detailed information about using the **spectrumscale** installation toolkit to configure GPFS and protocols, see the following:

- “spectrumscale command” on page 711
- Topic about using **spectrumscale** options to perform tasks in the *Installing GPFS on Linux nodes* chapter of *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- *Using the spectrumscale installation toolkit to perform installation tasks: Explanations and examples* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Chapter 5. Managing protocol data exports

You can manage the data exports that you have created using NFS, SMB, and Object.

Managing SMB shares

All SMB administration commands can be run from any cluster node including non-CES nodes. However, the latency of the administration command execution on a CES node is lower as the administrative changes can be applied straight away. Use the following information to manage SMB shares in IBM Spectrum Scale.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > SMB Shares**.

Creating SMB share

Use the following information to create an SMB share:

1. Create the directory to be exported through SMB:

```
mmscrfileset fs01 fileset --inode-space=new
mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
mkdir /gpfs/fs01/fileset/smb
```

Note: IBM recommends an independent fileset for SMB shares.

Create a new independent fileset with these commands:

```
mmscrfileset fs01 fileset --inode-space=new
mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
```

If the directory to be exported does not exist, create the directory first by running the following command:

```
mkdir /gpfs/fs01/fileset/smb"
```

2. The recommended approach for managing access to the SMB share is to manage the ACLs from a Windows client machine. To change the ACLs from a Windows client, change the owner of the share folder to a user ID that will be used to make the ACL changes by running the following command:

```
chown 'DOMAIN\smbadmin' /gpfs/fs01/fileset/smb
```

3. Create the actual SMB share on the existing directory:

```
mmsmb export add smbexport /gpfs/fs01/fileset/smb
```

Additional options can be set during share creation. For the documentation of all supported options, see “mmsmb command” on page 663.

4. Verify that the share has been created:

```
mmsmb export list
```

5. Access the share from a Windows client using the user ID that has been previously made the owner of the folder.

6. Right-click the folder in the Windows Explorer, open the **Security** tab, click **Advanced**, and modify the Access Control List as required.

Note: An SMB share can only be created when the ACL setting of the underlying file system is **-k nfsv4**. In all other cases, **mmsmb export create** will fail with an error.

See “Authorizing protocol users” on page 200 for details and limitations.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > SMB Shares**.

Changing SMB share configuration

Use the following information to change the SMB share configurations.

For the documentation of all supported options, see “mmsmb command” on page 663.

To see a list of supported configuration options for SMB shares, run the command:

```
mmsmb export change --key-info supported
```

For example, to change the descriptive comment for a share, run the command:

```
mmsmb export change smbshare --option 'comment=Project X export'
```

To list the configuration of all SMB shares, run the command:

```
mmsmb export list --all
```

Note: Changes to SMB share configurations only apply to client connections that have been established after the change has been made.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > SMB Shares**.

Creating SMB share ACLs

The SMB protocol supports a separate level of ACLs that can be optionally added to an SMB share.

For more information, see Managing ACLs of SMB exports using MMC.

For details, see the information about managing the SMB export ACLs from a Windows client through the MMC.

Removing SMB shares

To remove an SMB share, use the **mmsmb** command. Use the following information to remove SMB shares.

To remove an SMB share:

1. Run the following command:

```
mmsmb export remove smbexport
```

2. Verify that the export has been removed by listing the configured SMB share again:

```
mmsmb export list
```

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > SMB Shares**.

Listing SMB shares

To list the SMB shares, run the following command:

```
mmsmb export list
```


Managing SMB shares using MMC

Microsoft Management Console (MMC) is a Windows tool that can be used to do basic configuration tasks on an SMB server. These tasks include administrative tasks such as listing or closing the connected users and open files, and creating and manipulating SMB shares. You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for managing SMB shares on the IBM Spectrum Scale cluster.

Attention: Listing a large number of entities (thousands of files, connections, locks, etc.) using Microsoft Management Console (MMC) might take a very long time and it might impact the performance of the file server. In these cases, it is recommended to use server-side administration tools.

Ensure that the following tasks are complete before you manage SMB shares:

- IBM Spectrum Scale is installed and configured.
- The SMB protocol is enabled and healthy SMB services are running on all protocol nodes.
- Required SMB shares are created and mounted from the Windows client.
- Microsoft Active Directory (AD) based authentication is set up. This includes:
 - Cluster nodes and client are domain members.
 - The client on which Microsoft Management Console (MMC) is running is a domain member.
 - Accurate DNS information is configured. If active sessions are listed, MMC tries to do a reverse pointer record lookup with DNS for every session (client IP), and if that fails then MMC hangs.
 - Involved NetBIOS names can be resolved using DNS.

For using the Shared Folders Microsoft Management Console (MMC) snap-in, you must be a member of the local administrators group of the cluster. After joining the cluster to an AD domain, only the domain admins group is a member of the administrators group of the cluster.

To add other users who can use the Shared Folders Microsoft Management Console (MMC) snap-in:

1. Connect to MMC as a user that is a member of the domain admins group.
2. Navigate to **System Tools > Local Users and Groups** and add a user to the local administrators group.

For more information, see the Microsoft Management Console documentation.

The following MMC features are not supported for managing SMB shares on the IBM Spectrum Scale cluster:

- Audit of MMC read operations
- Event viewer
- Setting max connections per share

Connecting to SMB shares by using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for connecting to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.

The Shared Folders Microsoft Management Console (MMC) snap-in opens.

2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

Note: If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see “Managing SMB shares using MMC” on page 115.

Creating SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for creating SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:

- a. Click **Start > Run**.
- b. Type `fsmgmt.msc` and click **OK**.

The Shared Folders Microsoft Management Console (MMC) snap-in opens.

2. Connect to the server on which you want to create SMB shares:

- a. Click **Action > Connect to another computer**.
- b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.

3. In the left pane, right-click **Shares** and then click **New Share**. The Create A Shared Folder wizard opens.

Note: If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see “Managing SMB shares using MMC” on page 115.

4. In the Create A Shared Folder wizard, click **Next**.

5. In the **Folder path** field, enter the share path and click **Next**.

Note: The directory for the SMB has to already exist in the file system.

6. Enter the SMB share name and description, select the required offline setting, and then click **Next**.

7. Select the required SMB share permission setting and click **Finish**.

Modifying or removing SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for modifying or removing SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:

- a. Click **Start > Run**.
- b. Type `fsmgmt.msc` and click **OK**.

The Shared Folders Microsoft Management Console (MMC) snap-in opens.

2. Connect to the server on which you want to create SMB shares:

- a. Click **Action > Connect to another computer**.
- b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.

3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

Note: If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see “Managing SMB shares using MMC” on page 115.

4. Do one of the following steps depending on whether you want to modify or remove SMB shares:

- To modify an SMB share:
 - a. In the right pane, right-click the SMB share that you want to modify, and then click **Properties**.
 - b. Modify the properties as required and click **OK**.
- To remove an SMB share:

- a. In the right pane, right-click the SMB share that you want to remove, and then click **Stop Sharing**.

Managing ACLs of SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for managing access control lists (ACLs) of SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.

The Shared Folders Microsoft Management Console (MMC) snap-in opens.

2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

Note: If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see “Managing SMB shares using MMC” on page 115.

4. In the right pane, right-click the SMB share for which you want to view or change the permissions and then click **Properties**.
5. You can do one of the following:
 - To view the permissions a user or a group has for the SMB share, on the **Share Permissions** tab, under the "Group or user names" pane, click on the user name or the group name.
The permissions are displayed in the "Permissions for" pane.
 - To change the permissions a user or a group has for the SMB share, on the **Security** tab, under the "Group or user names" pane, click on the user name or the group name and then click **Edit**.

Note: Changes affect only the SMB share, not the ACL in the file system of the exported directory. For information on permissions that you can change, see documentation for the Shared Folders Microsoft Management Console (MMC) snap-in.

Modifying offline settings of SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for modifying offline settings of SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.
- The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
 3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

Note: If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see “Managing SMB shares using MMC” on page 115.

4. In the right pane, right-click the SMB share whose offline settings you want to modify, and then click **Properties**.

5. On the **General** tab, click **Offline Settings**.
6. In the Offline Settings window, configure the offline settings of the SMB share. For information on offline settings that you can configure, see documentation for the Shared Folders Microsoft Management Console (MMC) snap-in.

Viewing active connections to SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing active connections to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Sessions**. All active connections to SMB shares are listed in the right pane.

Disconnecting active connections to SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for disconnecting active connections to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Sessions**. All active connections to SMB shares are listed in the right pane.
4. In the right pane, right-click the connection that you want to close and then click **Close Session**.

Attention: If connections are forced to close, data loss might occur for open files on the connections getting closed.
5. Click **OK** to confirm.

Viewing open files in SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing open files in SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
 - a. Click **Start > Run**.
 - b. Type `fsmgmt.msc` and click **OK**.The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
 - a. Click **Action > Connect to another computer**.
 - b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Open Files**. All open files in SMB shares are listed in the right pane.

Viewing locks on files in SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing locks on open files in SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:

- a. Click **Start > Run**.
- b. Type `fsmgmt.msc` and click **OK**.

The Shared Folders Microsoft Management Console (MMC) snap-in opens.

2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:

- a. Click **Action > Connect to another computer**.
- b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.

3. In the left pane, click **Open Files**. All open files in SMB shares are listed in the right pane.

4. In the right pane, view locks on a file under the **# Locks** column.

The number of locks is displayed under the **# Locks** column and the type of locks is displayed under the **Open Mode** column.

SMB share limitations

When you create SMB shares, consider their limitations and support restrictions.

- NTFS alternate data streams are not supported. For example, named streams generated by a Mac OS X operating system cannot be stored directly.
- The encryption status of files cannot be queried or changed from SMB clients. Use the **mm CLI** commands instead.
- When propagation of opportunistic locks across protocols is enabled (SMB option `gpfs:leases`), then Level 2 oplocks are not granted and Exclusive or batch oplocks are not broken down to Level 2 oplocks and are revoked from the system.
- Symbolic links cannot be created or changed from SMB clients and are not reported as symbolic links.
- Symbolic links created via NFS or directly in the file system will be respected as long as they point to a target under the same shared directory.
- Distributed File System (DFS) is not supported.
- Windows Internet Name Service (WINS) is not supported.
- Retrieving Quota information using `NT_TRANSACT_QUERY_QUOTA` is not supported.
- Setting Quota information using `NT_TRANSACT_SET_QUOTA` is not supported.
- Setting the maximum number of connections to a share is not supported. The MMC GUI allows specifying this parameter, but it cannot be set on an IBM Spectrum Scale cluster.
- Unix Extensions are not supported.
- You cannot create a shadow copy of a shared folder using a remote procedure call from a shadow copy client. Backup utilities, such as Microsoft Volume Shadow Copy Service, cannot create a shadow copy of a shared folder using a procedure call.
- The Branch Cache hash operations using `SRV_READ_HASH_IOCTL` are not supported.
- Leases are not supported.
- Only the SMB2 and SMB3 protocol versions are supported.
- Only mandatory SMB3 protocol features are supported.
- No support of dynamic ACLs and SACLs.
- No support of SID history.
- No support of SMB 3.1.

SMB Clients:

- Windows: SMB 1 is not supported, thus no access for Windows XP clients

- Linux: SMB1 is not supported, so make sure to use the version option with the kernel SMB client:
`mount.cifs //fscs-p8-11/ralph /media/ss -o user=aduser1,pass=Passw0rd,dom=W2K8DOM05,vers=2.0`

Managing NFS exports

Use the following information to manage NFS exports in IBM Spectrum Scale.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > NFS Exports**.

Creating NFS exports

To add an NFS export, use the **mmnfs** export add command.

1. If the directory to be exported does not exist, create the directory by running the following commands:

```
mmcrfileset fs01 fileset --inode-space=new
mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
```

For more details, see “mmcrfileset command” on page 408 and “mmlinkfileset command” on page 517.

Note: We recommend an independent fileset for NFS exports.

2. Adjust the ownership and permissions of the folder as required.
Use the GPFS ACL's with **mmgetacl** and **mmputacl** to set the correct ownership and the access permission.
Additional options can be set during export creation. For the documentation of all supported options, see “mmnfs command” on page 570.
3. Create the NFS export using the following command:
`mmnfs export add /gpfs/fs01/fileset`

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols > NFS Exports**.

Changing NFS export configuration

After an NFS export is created, the export attributes can be changed by using the **mmnfs export change** command.

For the documentation of all supported options, see “mmnfs command” on page 570.

- l For example, to grant another client IP address access to the NFS export, run the following command:
`mmnfs export change /gpfs/fs01/nfs --nfsadd 10.23.23.23`

After the change is made, verify the configuration by running the following command:

```
mmnfs export list
```

- l Changing the NFS export configuration is not dynamic; NFS services automatically restart during the **mmnfs export change** command. This means that an NFS client that did a soft mount might lose connectivity, which might result in application failures; an NFS client that did a hard mount might "stall" during the NFS grace period.

Removing NFS exports

To remove an NFS export, use the `mmnfs export remove` command.

To remove an NFS export:

1. Run the following command:
`mmnfs export remove /gpfs/fs01/nfs`
2. Verify that the export has been removed by listing the configured NFS exports again:
`mmnfs export list`

Listing NFS exports

To list the NFS exports, run the following command:

```
mmnfs export list
```

Multiprotocol exports

Exports for SMB and NFS protocols can be configured so that they have access to the same data in the GPFS file system.

To export data via multiple protocols, first create an export for one protocol using the appropriate GPFS command (for example, `mmnfs export add`). In order to export the same GPFS path via a second protocol, simply create another export using the protocol-specific export management command (for example, `mmsmb export add`).

The operations of adding and removing exports do not delete any data in the GPFS file system. If at a later time access to a GPFS file system for a specific protocol needs to be removed, this can be done via the corresponding command. Removal of exports is a logical operation and does not change the data in the GPFS file system. It also does not impact access to the same data configured via another protocol.

Multiprotocol export considerations

Exports for SMB and NFS protocols can be configured so that they have access to the same data in the file system. In addition, the data can be accessed directly in the file system on the cluster nodes. When configuring access to the same GPFS file system via both the NFS and SMB protocols, certain limitations apply.

These restrictions apply to the general areas of file locking (including share reservation and lock semantics), recovery (reclaim), and cross-protocol notifications.

SMB in IBM Spectrum Scale can be configured to maintain locks in SMB rather than GPFS. In this mode NFS is not aware of SMB locks and therefore it should not be used with concurrent access via NFS.

Furthermore, SMB is not aware of NFS grace periods (in which NFS clients are given time to reclaim any locks and share reservations). If you expect a lot of contention between SMB and NFS, NFSv4 reclaims might fail.

The following additional limitations apply:

- The SMB exports should be configured with `gpfs:leases=yes` and `gpfs:sharemodes=yes`. This implies that no level 2 oplocks will be granted to SMB clients.
- The NFS server always relies on the GPFS file system for managing file locks and uses the GPFS internal mechanism to synchronize these locks across all the nodes in the cluster. The NFS server is informed by GPFS of any changes on the file system objects, so NFS clients when obtaining file system information always are presented with the latest status.

Chapter 6. Managing object storage

Use the following information to use and manage the IBM Spectrum Scale for object storage features.

Understanding and managing Object services

Use the following information to manage services related to IBM Spectrum Scale for object storage.

IBM Spectrum Scale uses the **mmces service** command to enable, start, stop, or disable Object services on all protocol nodes.

The enable and disable operations are cluster-wide operations. To enable or disable the Object protocol, use **mmces service [enable | disable] OBJ**. The Object protocol must have been initially configured using the **mmobj swift base** command before it can be enabled in the cluster.

CAUTION:

Disabling the object service unconfigures the Object protocol and discards OpenStack Swift configuration and ring files from the CES cluster. If Openstack Keystone configuration is configured locally, disabling object storage also discards the Keystone configuration and database files from the CES cluster. However, to avoid accidental data loss, the associated filesets used for the object data are not automatically removed during disable. The filesets for the object data and any filesets created for optional object storage policies need to be removed manually. For enabling the object service subsequently, either different fileset names need to be specified or the existing filesets need to be cleaned up. For information on cleaning up the object filesets, see the steps "Remove the fileset created for object" and "Remove any fileset created for an object storage policy" (if applicable) in the Cleanup procedures required if reinstalling with the spectrumscale installation toolkit topic of IBM Spectrum Scale: Concepts, Planning, and Installation Guide.

Note: To disable the object protocol, first remove the object authentication. For complete usage information, see the "mmuserauth command" on page 690.

In addition, enabled Object service can be started and stopped on individual nodes or cluster-wide.

To start or stop the Object protocol cluster-wide, use **-a** flag **mmces service [start | stop] OBJ -a**.

To start or stop the Object protocol on individual nodes, use **mmces service [start | stop] OBJ -N <node>**.

Attention: If object services on a protocol node are stopped by the administrator manually, access to object data might be impacted unless the CES IP addresses are first moved to another node. There are multiple ways to accomplish this, but the simplest is to suspend the node. After suspending a node, CES automatically moves the CES IPs to the remaining nodes in the cluster. However, doing this suspends operation for all protocols running on that protocol node.

If you want to stop object services on a protocol node, you can use the following steps:

1. Suspend CES operations on the protocol node using the **mmces node suspend** command.
2. View the CES IP addresses on that node using the **mmces address list** command and verify that all CES IP addresses have been moved to other protocol nodes.
3. Stop the object services using the **mmces service stop OBJ** command.

Performing these steps ensures that object functionality is available on other nodes in the cluster.

To restore object services on that protocol node, you can use the following steps:

1. Resume CES operations on the protocol node using the **mmces node resume** command.
2. View the CES IP addresses on that node using the **mmces address list** command and verify that all CES IP addresses have been moved to that protocol node.
3. Start the object services using the **mmces service start OBJ** command.

Use the **mmces service list** command to list the protocols enabled on IBM Spectrum Scale. List a verbose output of object services running on the local node using the **-v** flag as shown in the following example:

```
# mmces service list -v
Enabled services: OBJ SMB NFS
OBJ is running
OBJ:openstack-swift-object          is running
OBJ:openstack-swift-account         is running
OBJ:openstack-swift-container       is running
OBJ:openstack-swift-proxy           is running
OBJ:memcached                       is running
OBJ:openstack-swift-object-replicator is running
OBJ:openstack-swift-account-reaper  is running
OBJ:openstack-swift-account-replicator is running
OBJ:openstack-swift-container-replicator is running
OBJ:openstack-swift-object-sof      is running
OBJ:httpd (keystone)               is running
SMB is running
NFS is running
```

For complete usage information, see “mmces command” on page 304.

Every object protocol node can access every virtual device in the shared file system, and some OpenStack Swift object services can be optimized to take advantage of this by running from a single Object protocol node.

Even though objects are not replicated by OpenStack Swift, the **swift-object-replicator** runs to periodically clean up tombstone files from deleted objects. It is run on a single Object protocol node and manages cleanup for all of the virtual devices.

The **swift-object-updater** is responsible for updating container listings with objects that were not successfully added to the container when they were initially created, updated, or deleted. Like the object replicator, it is run on a single object protocol node.

The following table shows each of the object services and the set of object protocol nodes on which they need to be executed.

Table 8. Object services and object protocol nodes

Object service	GPFS protocol node
ibmobjectizer	object-singleton_node ¹
openstack-swift-account	All
openstack-swift-account-auditor	object_singleton_node
openstack-swift-account-reaper	All
openstack-swift-account-replicator	All
openstack-swift-container	All
openstack-swift-container-auditor	object_singleton_node
openstack-swift-container-updater	object_singleton_node
openstack-swift-container-replicator	All
openstack-swift-object	All

Table 8. Object services and object protocol nodes (continued)

Object service	GPFS protocol node
<code>openstack-swift-object-auditor</code>	<code>object_singleton_node</code> ²
<code>openstack-swift-object-replicator</code>	All
<code>openstack-swift-object-sof</code>	All ¹
<code>openstack-swift-object-updater</code>	<code>object_singleton_node</code>
<code>openstack-swift-object-expirer</code>	<code>object_singleton_node</code>
<code>openstack-swift-proxy</code>	All
<code>memcached</code>	All
<code>openstack-keystone</code>	All ^{3, 4}
<code>postgresql-obj</code>	<code>object_database_node</code> ³

¹ If unified file and object access is enabled.
² If multi-region object deployment is enabled.
³ If local OpenStack Keystone Identity Service is configured.
⁴ Updated to `httpd (keystone)` on all nodes, if using local authentication.

Understanding the mapping of OpenStack commands to IBM Spectrum Scale administrator commands

Use this information to map OpenStack commands to IBM Spectrum Scale administrator commands.

In IBM Spectrum Scale, for Object storage, several OpenStack commands have been replaced with IBM Spectrum Scale commands for easy maintenance. This section identifies those commands.

1. Ring Building

The `swift-ring-builder` command should only be used to view the object, container, and account ring on any IBM Spectrum Scale protocol node. The user should not directly execute any commands that modify the ring. All ring maintenance operations are handled automatically by the CES infrastructure.

For example, when a new CES IP address is added to the configuration, all rings are automatically updated to distribute Swift virtual devices evenly across CES IP addresses.

The master copy of each ring builder file is kept in the IBM Spectrum Scale Cluster Configuration Repository (CCR). Changes made locally to the ring files will be overwritten with the master copy when monitoring detects a difference between the ring file in CCR and the file in `/etc/swift`.

2. Configuration Changes

The `openstack-config` command should not be used to update any of the configuration files consumed by IBM Spectrum Scale for Object storage. Furthermore, you should not edit these files directly, but instead modify them using the `mmobj config change` command.

The master copy of object and related configuration files are kept in the IBM Spectrum Scale CCR. Changes made locally to these config files will be overwritten with the master copy when monitoring detects a difference between the configuration file in CCR and the file in `/etc/swift` or `/etc/keystone`.

Changing Object configuration values

Use the following information to change the Object configuration values in the Cluster Configuration Repository (CCR).

You can manage the Object configuration data in the Cluster Configuration Repository (CCR). When an object configuration file is changed, callbacks on each protocol node will update that node with the change and restart one or more Object services if necessary.

To change the Object configuration, use the `mmobj` command so that the change is made in the CCR and propagated correctly across the Swift cluster.

For more details, see the “`mmobj` command” on page 581.

Changing the object base configuration to enable S3 API emulation

IBM Spectrum Scale uses Swift3 Middleware for OpenStack Swift, allowing access to IBM Spectrum Scale using the Amazon Simple Storage Service (S3) API.

Note: In the following commands, if you add more than one value to a property, you must put single quotes around the values. For example:

```
mmobj config change --ccrfile proxy-server.conf --section filter:keystoneauth
--property operator_roles --value 'admin,SwiftOperator'
```

Perform the following steps if S3 API emulation was not enabled as part of the object base configuration:

1. Alter your `keystone-paste.ini`.

a. Alter pipeline:admin_api pipeline:

Run command:

```
mmobj config list --ccrfile keystone-paste.ini --section pipeline:admin_api
```

Was:

```
[pipeline:admin_api]
pipeline = sizelimit url_normalize request_id build_auth_context token_auth
json_body ec2_extension crud_extension admin_service
```

Run command:

```
mmobj config change --ccrfile keystone-paste.ini --section pipeline:admin_api --property
pipeline --value 'sizelimit url_normalize request_id build_auth_context token_auth
json_body ec2_extension s3_extension crud_extension admin_service'
```

Changed To:

```
[pipeline:admin_api]
pipeline = sizelimit url_normalize request_id build_auth_context token_auth
json_body ec2_extension s3_extension crud_extension admin_service
```

b. Alter pipeline:api_v3

Run command:

```
mmobj config list --ccrfile keystone-paste.ini --section pipeline:api_v3
```

Was:

```
[pipeline:api_v3]
pipeline = sizelimit url_normalize request_id build_auth_context token_auth
json_body ec2_extension_v3 simple_cert_extension revoke_extension federation_extension
oauth1_extension endpoint_filter_extension endpoint_policy_extension service_v3
```

Run command:

```
mmobj config change --ccrfile keystone-paste.ini --section pipeline:api_v3 --property pipeline --value
'sizelimit url_normalize request_id build_auth_context token_auth json_body ec2_extension_v3 s3_extension
simple_cert_extension revoke_extension federation_extension oauth1_extension endpoint_filter_extension
endpoint_policy_extension service_v3'
```

:

Changed To

```
[pipeline:api_v3]
pipeline = sizelimit url_normalize request_id build_auth_context token_auth json_body
ec2_extension_v3 s3_extension simple_cert_extension revoke_extension federation_extension
oauth1_extension endpoint_filter_extension endpoint_policy_extension service_v3
```

```

| c. Add filter:s3_extension
| Run command:
| mmobj config list --ccrfile keystone-paste.ini --section filter:s3_extension
| Was:
| Section not found: filter:s3_extension
| Run command:
| mmobj config change --ccrfile keystone-paste.ini --section filter:s3_extension
| --property paste.filter_factory --value keystone.contrib.s3:S3Extension.factory
| Change to:
| [filter:s3_extension]
| paste.filter_factory = keystone.contrib.s3:S3Extension.factory
| 2. Alter your proxy-server.conf
| a. Alter pipeline:main pipeline property
| Run command:
| mmobj config list --ccrfile proxy-server.conf --section pipeline:main
| Was:
| [pipeline:main]
| pipeline = healthcheck cache formpost tempurl authtoken keystone container-quotas
| account-quotas staticweb bulk slo dlo proxy-server
| Run command:
| mmobj config change --ccrfile proxy-server.conf --section "pipeline:main" --property pipeline
| --value 'healthcheck cache formpost tempurl swift3 s3token authtoken keystoneauth
| container-quotas account-quotas staticweb bulk slo dlo proxy-server'
| Change To:
| [pipeline:main]
| pipeline = healthcheck cache formpost tempurl swift3 s3token authtoken keystoneauth
| container-quotas account-quotas staticweb bulk slo dlo proxy-server
| b. Add swift3 filter
| Run command:
| mmobj config list --ccrfile proxy-server.conf --section filter:swift3
| Was:
| Section not found: filter:swift3
| Run the following commands. Each command is on one line of the console:
| mmobj config change --ccrfile proxy-server.conf --section filter:swift3
| --property use --value egg:swift3#swift3
| mmobj config change --ccrfile proxy-server.conf --section filter:swift3
| --property s3_acl --value true
| mmobj config change --ccrfile proxy-server.conf --section filter:swift3
| --property dns_compliant_bucket_names --value false
| Changed To:
| [filter:swift3]
| use = egg:swift3#swift3
| s3_acl = true
| dns_compliant_bucket_names = false
| c. Alter filter:keystone to filter:keystoneauth
| Run command:
| mmobj config list --ccrfile proxy-server.conf --section filter:keystoneauth
|
| If there are properties listed and they are what you expect, you are done with this step.
| If the filter:keystoneauth section does not exist, issue this command to list the entries under the
| filter:keystone section:
| mmobj config list --ccrfile proxy-server.conf --section filter:keystone
|
| For each entry listed, add it to the filter: keystoneauth section.

```

```

| For example,
| mmobj config change --ccrfile proxy-server.conf --section filter:keystoneauth --property use --value egg:swift#keystoneauth
|
| Repeat this for each property/value pair listed.
| d. Add s3token filter
| Run command:
| mmobj config list --ccrfile proxy-server.conf --section filter:s3token
| Was:
| Section not found: filter:s3token
| Run commands:
| mmobj config change --ccrfile proxy-server.conf --section filter:s3token --property auth_host --value 127.0.0.1
| mmobj config change --ccrfile proxy-server.conf --section filter:s3token --property auth_port --value 35357
| mmobj config change --ccrfile proxy-server.conf --section filter:s3token --property auth_protocol --value http
| mmobj config change --ccrfile proxy-server.conf --section filter:s3token --property paste.filter_factory
| --value keystonemiddleware.s3_token:filter_factory
|
| Changed To:
| [filter:s3token]
| auth_host = 127.0.0.1
| auth_port = 35357
| auth_protocol = http
| paste.filter_factory = keystonemiddleware.s3_token:filter_factory

```

Configuring OpenStack EC2 credentials

The credentials that are used on the Amazon S3 and Elastic Compute Cloud (EC2) APIs are different from the credentials that are used by the OpenStack API. As a result, you must generate these special credentials to use them when accessing the IBM Spectrum Scale OpenStack services.

The credentials are created by the OpenStackClient, a command-line client for OpenStack, that allows the creation and use of access/secret pairs for a user/project pair. This requires the operators to create the access/secret for each user/project pair.

1. Source openrc with the admin credentials.
2. Create EC2 credential by running this command:

```
openstack credential create --type ec2 --project <project> <user> '{"access": <aws_access_key>, "secret": <aws_secret_key>}'
```

Note: Ensure to use Keystone UUIDs rather than names if duplicate user/project names exist across domains. Additionally, the admin users should be able to list and delete access/secrets for a specific user/project.

3. View all EC2 credentials by running this command:

```
openstack credential list
openstack credential show <credential-id>
```

4. You can change your Access Key ID and Secret Access Key if necessary.

It is recommended to have regular rotation of these keys and switching applications to use the new pair.

Change the EC2 credentials by running this command:

```
openstack credential set --type ec2 --data '{"access": <access>, "secret": <secret>}' --project <project> <credential-id>
```

5. Delete the EC2 credentials by running this command:

```
openstack credential delete <credential_id>
```

The following example shows the creation of EC2 credentials using the admin project and the admin user IDs:

Where openrc contains:

```
export OS_AUTH_URL="http://127.0.0.1:35357/v3"
export OS_IDENTITY_API_VERSION=3
```

```

export OS_AUTH_VERSION=3 export
OS_USERNAME="admin"
export OS_PASSWORD="Passw0rd"
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_NAME=Default
source openrc

```

```
openstack credential create --type ec2 --project admin admin '{"access": "022AB06E7MXB9H9DM02", "secret": "pWcu1UX4JEDGM/LtmEENI/aVmYvHniF5zB+d9+ct"}'
```

You are now ready to connect to the IBM Spectrum Scale Object store using the Amazon S3 API. You can connect with any S3-enabled client.

Managing OpenStack access control lists using S3 API emulation

Use the following information to manage OpenStack ACLs using S3 API emulation..

IBM Spectrum Scale supports S3 access control lists (ACLs) on buckets and objects. These S3 ACLs are stored separately from the ACLs set through the Swift API and the ACLs stored in the file system (NFSv4 or POSIX). For information on how to set and query ACLs through the S3 API, see the Amazon S3 documentation.

The following information about S3 ACLs is only applicable if **s3_acl** is set to true in the `proxy-server.conf` file. If S3 API emulation enabled, **s3_acl** is set to true by default.

For a user to use the S3 API in IBM Spectrum Scale, the user must have a role defined for the swift project. Any role suffices, because for the S3 API there is no difference between the `SwiftOperator` role or others.

The following table lists the required permissions for S3 operations.

S3 operation	Required permission
PUT object	WRITE permission on bucket or as bucket owner
HEAD object	READ permission on object or as object owner
GET object	READ permission on object or as object owner
DELETE object	WRITE permission on bucket or as bucket owner
Get object ACL (GET on ACL subresource)	READ_ACP permission on object or as object owner
Set object ACL (PUT on ACL subresource)	WRITE_ACP permission on object or as object owner
Create bucket (PUT)	Any user with a role on the project can create a bucket.
HEAD bucket	READ permission on bucket or as bucket owner
GET bucket	READ permission on bucket or as bucket owner
DELETE bucket	bucket owner
Get bucket ACL (GET on ACL subresource)	READ_ACP permission on bucket or as bucket owner
Set bucket ACL (PUT on ACL subresource)	WRITE_ACP permission on bucket or as bucket owner

Known limitations for S3 API emulation support

- Unauthorized S3 requests are not supported. S3 requests do not contain a reference to the account, and the object server derives the account information from the authorization information. This is not possible for unauthorized requests.
- Specifying S3 ACL grantees by email is not supported.

- Grantees in ACL are not validated. Therefore, any name can be used, even users that do not exist.
- The owner of a resource is implicitly granted **FULL_CONTROL** instead of just **READ_ACP** and **WRITE_ACP**. This is not a security issue because with **WRITE_ACP**, the owners can grant themselves **FULL_CONTROL** access.
- Container or objects created using the swift API are not accessible through the S3 API when the configuration flag `allow_no_owner` is set to `false` in `proxy-server.conf`. To change this setting, you can use the following command:


```
mmobj config change --ccrfile proxy-server.conf --section filter:swift3
--property allow_no_owner --value true
```
- The POST operation to update metadata has not been implemented.

Managing object capabilities

You can manage the object capabilities using the following commands.

For an overview of object capabilities, see *Object capabilities* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

- To list all object capabilities available cluster wide, use the **mmobj config list** command as follows:

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
```

The system displays output similar to the following:

```
file-access-enabled: true
multi-region-enabled: true
s3-enabled: false
```

You can also list specific object capabilities using the **mmobj config list** command as follows:

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property multi-region-enabled
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property s3-enabled
```

- To enable the file-access object capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled --value true
```

The system displays output similar to the following:

```
file-access-enabled: true
```

- To disable the file-access capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled --value false
```

The system displays output similar to the following:

```
file-access-enabled: false
```

Mapping of storage policies to filesets

For every storage policy created using the **mmobj policy create** command, one fileset is created or reused.

After a storage policy is created, you can specify that storage policy while creating new containers to associate that storage policy with those containers. When objects are uploaded into a container, they are stored in the fileset that is associated with the container's storage policy. For every new storage policy, a new object ring is created. The ring defines where objects are located and also defines multi-region replication settings.

The name of the fileset can be specified optionally as an argument of the **mmobj policy create** command. An existing fileset can be used only if:

- It is not being used for an existing storage policy.
- It is a part of the object file system
- Its junction path is not nested to other filesets.

If even one of these prerequisites is missing, the **mmobj policy create** command fails. Otherwise, the fileset is used and the softlinks for the devices that are given to the ring builder point to it. If no fileset name is specified with the **mmobj policy create** command, a fileset is created using the policy name as a part of the fileset name with the prefix `obj_`.

For example, if a storage policy with name `Test` is created and no fileset is specified, a fileset with the name `obj_Test` is created and is linked to the base file system for object:

```
<object base filesystem mount point>/obj_Test/<n virt. Devices>
```

Attention: For any fileset that is created, its junction path is linked under the base file system for object. Any fileset that is used for a storage policy and its corresponding softlinks must not be changed for the junction path and the soft links. If it is changed, data might be lost or it might get corrupted.

To enable swift to work with the fileset, softlinks under the given devices path in `object-server.conf` are created:

```
<devices path in object-server.conf>/<n virt. Devices>
<object base filesystem mount point>/obj_Test/<n virt. Devices>
```

Administering storage policies for object storage

Use the following information to create, list, and change storage policies for object storage.

Before creating a storage policy with the file-access (unified file and object access) function enabled, the file-access object capability must be enabled. For more information, see *Object capabilities in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*, and *Managing object capabilities in IBM Spectrum Scale: Administration and Programming Reference*.

- To create a new storage policy with the unified file and object access feature enabled, use the following command:

```
mmobj policy create sof-policy --enable-file-access
```

The system displays output similar to this:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_sof-policy
[I] Creating new unique index and build the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

- To list storage policies for object storage with details of functions available with those storage policies, use the following command:

```
mmobj policy list --verbose
```

The system displays output similar to this:

Index	Name	Deprecated	Fileset	Fileset Path	Functions	Function Details
0	SwiftDefault		object_fileset	/ibm/cesSharedRoot/object_fileset		
11751509160	sof-policy		obj_sof-policy	/ibm/cesSharedRoot/obj_sof-policy	file-and-object-access	regions="1"
11751509230	mysofpolicy		obj_mysofpolicy	/ibm/cesSharedRoot/obj_mysofpolicy	file-and-object-access	regions="1"

- To change a storage policy for object storage, use the following command:

```
mmobj policy change
```

For more information about the **mmobj policy** command, see “`mmobj command`” on page 581.

Creating storage policy for object compression

Use the following information to create a storage policy with the compression function enabled and to create a storage policy with the compression schedule defined.

- To create a storage policy with the compression function enabled, use the `--enable-compression` option with the **mmobj policy create** command as follows:

```
mmobj policy create CompressionTest --enable-compression --compression-schedule "MM:HH:dd:ww"
```

- To create a storage policy with the compression function enabled and a compression schedule defined, use the `--enable-compression` and the `--compression-schedule` options with the **mmobj policy create** command as follows:

```
mmobj policy create CompressionTest --enable-compression --compression-schedule "MM:HH:dd:ww"
```

where

MM = 0-59 minutes

HH = 0-23 hours

dd = 1-31 day of month

ww = 0-7 (0=Sun, 7=Sun) day of week

- Use * for specifying every instance of a unit. For example, `dd = *` means that the job is scheduled to run every day.
- Comma separated lists are allowed. For example, `dd = 1,3,5` means that the job is scheduled to run on every 1st, 3rd, 5th of a month.
- If `ww` and `dd` both are specified, the union is used.
- Specifying a range using `-` is not supported.
- Empty values are allowed for `dd` and `ww`. If empty, `dd` and or `ww` are not considered.
- Empty values for `mm` and `hh` are treaded as `*`.

In the following example, the compression job has been scheduled to run at 23.50 every day:


```
mmobj policy create CompressionTest --enable-compression --compression-schedule "50:23:*:*"
```

Every object stored using a storage policy that has compression enabled is compressed according to the specified schedule. There is no need to decompress an object in advance of a get request or any other object request. IBM Spectrum Scale automatically returns the decompressed object.

Note: The download performance of objects in a compressed container is reduced compared to the download performance of objects in a non-compressed container.

Note: The same compression functionality and restrictions apply to object compression and file compression.

Related concepts:

|  File compression

| You can compress or decompress files either with the **mmchattr** command or with the **mmapplypolicy** command with a **MIGRATE** rule. You can do the compression or decompression synchronously or defer it until a later call to **mmrestripefile** or **mmrestripefs**.

Adding a region in a multi-region object deployment

Perform the following steps to add a region in a multi-region object deployment environment.

In the command examples, Europe is the first region and Asia is the second region.

1. Export the information of the first region to a file by using the **mmobj multiregion export** command.

For example:

```
[europe]# mmobj multiregion export --region-file /tmp/multiregion_europe.dat
```

2. Copy the file manually to the second region.

For example:

```
[europe]# scp /tmp/multiregion_europe.dat asia:/tmp
```

3. From the second region, join the multi-region environment as follows:
 - a. Use the file generated in the first region while deploying object on the second region by using the **mmobj swift base** command.

For example:

```
[asia]# mmobj swift base -g /mnt/gpfs0 --cluster-hostname gpfs-asia --admin-password Passw0rd  
-i 100000 --admin-user admin --enable-s3 \  
--enable-multi-region --remote-keystone-url http://gpfs-asia:35357/v3 \  
--join-region-file /tmp/multiregion_europe.dat \  
--region-number 2 --configure-remote-keystone
```

This step installs the object protocol in the 2nd region and joins the 1st region. Additional devices are added to the primary ring files for this region.

4. Export the ring file data of the second region.

For example:

```
[asia]# mmobj multiregion export --region-file /tmp/multiregion_asia.dat
```

5. Copy the file manually to the first region.

For example:

```
[asia]# scp /tmp/multiregion_asia.dat europe:/tmp
```

6. In the first region, update the local ring files with the configuration of the second region.

For example:

```
[europe]# mmobj multiregion import --region-file /tmp/multiregion_asia.dat
```

This step reads in the ring files which are updated with the information of the second region. This update ensures that the data of the second region contains a new region and therefore replaces the associated ring files in the first region with the ones from the second region.

Note:

Now the two clusters have been synced together and can be used as a multi-region cluster. Objects can be uploaded and downloaded from either region. If the installation of the second region specified the **--configure-remote-keystone** flag, a region-specific endpoint for the object-store service for the 2nd region is created in Keystone.

The regions need to be synced in the future any time region-related information changes. This includes changes in the set of CES IP addresses (added or removed) or if storage policies were created or deleted within a region. Changes that affect the `swift.conf` file or ring files need to be synced to all regions. For example, adding additional CES addresses to a region causes the ring files to be rebuilt.

7. In the second region, add CES addresses and update other clusters.

For example:

```
[asia]# mmces address add --ces-ip asia9
```

This step adds an address to the CES IP pool. This also triggers a ring rebuild which changes the IP-to-device mapping in the ring files.

8. Export the ring data so the other clusters in the region can be updated with the new IPs from the second region.

For example:

```
[asia]# mmobj multiregion export --region-file /tmp/multiregion_asia.dat
```

9. Copy the file manually to the first region.

For example:

```
[asia]# scp /tmp/multiregion_asia.dat europe:/tmp
```

10. In the first region, update with changes for the new second region address in the ring.

For example:

```
[europe]# mmobj multiregion import --region-file /tmp/multiregion_asia.dat
```

This step imports the changes from the second region. When this is complete, a checksum is displayed which can be used to determine when regions are synchronized together. By comparing it to the one printed when the region data was exported, you can determine that the regions are synchronized when they match. In some cases, the checksums do not match after import. This is typically due to some local configuration changes on this cluster which are not yet synced to the other regions. If the checksums do not match, then this region's configuration needs to be exported and imported into the other region to sync them.

Administering a multi-region object deployment environment

Use the following information to administer a multi-region object deployment environment.

A multi-region environment consists of several independent storage clusters linked together to provide unified object access. Configuration changes in one cluster which affect the multi-region environment are not automatically distributed to all clusters. The cluster which made the configuration change must export the relevant multi-region data and then the other regions must import that data to sync the multi-region configuration. Changes which affect multi-region are:

- Changes to the CES IP pool, such as adding or deleting addresses, which affect the ring layout.
- Changes to the object services ports used for the account, container, and object servers (ports 6200-6202).
- Creation, deletion, or modification of storage policies.
- Changes to the `swift.conf` configuration file

Use the following commands to manage the configuration of the multi-region environment:

- To export the data for the current region so that it can be integrated into other regions, use the following command. The *RegionData* file created can be used to update other regions:

```
mmobj multiregion export --region-file RegionData
```

The *RegionData* file is created and it contains the updated multi-region information.

- To import the multi-region data to sync the configuration, use the following command. The *RegionData* must be the file created from the **mmobj multiregion export** command:

```
mmobj multiregion import --region-file RegionData
```

As part of the export/import commands, a region checksum is printed. This checksum can be used to ensure that the regions are in sync. If the checksums match, then the multi-region configuration of the clusters match. In some cases, the checksums do not match after import. This is because the cluster performing the import had local configuration changes which had not been synced with the other regions. For example, a storage policy was created but the multi-region configuration was not synced with the other regions. When this happens the import command prints a message that the regions are not fully in sync because of the local configuration and that the region data must be exported and imported to the other regions. Once all regions have matching checksums, the multi-region environment is in sync.

An existing region can be completely removed from the multi-region environment. This action permanently removes the region configuration, and the associated cluster cannot rejoin the multi-region environment.

The cluster of the removed region needs to disable object services since it will not be usable as a standalone object deployment.

- To remove a previously defined region from the configuration, use the following command:

```
mmobj multiregion remove --remove-region-number RegionNumber
```

The remove command must be run from a different region than the one being removed. The cluster associated with the removed region must cleanup object services as appropriate with the **mmces service disable OBJ -a** command to uninstall object services.

- You can display the current multi-region information using the following command:

```
mmobj multiregion list
```

Unified file and object access in IBM Spectrum Scale

Unified file and object access allows use cases where you can access data using object as well as file interfaces. Use the following information to manage unified file and object access including identity management modes for unified file and object access, authentication for unified file and object access, and objectization service schedule.

Important: In a unified file and object access environment, when objects are accessed from the object interface, file ACLs are not honored and vice versa.

For example: If user Bob ingests a file from the SMB interface and user Alice does not have access to that file from the SMB interface, it does not mean that Alice does not have access to the file from the object interface. The access rights of Alice for that file or object from the object interface depends on the ACL defined for Alice on the container in which that file or object resides.

Identity management modes for unified file and object access

The following section gives information about the two identity management modes for unified file and object access: local mode and unified mode. This section also describes how to configure these modes for a system.

Unified file and object access comprises the following two modes:

- **local_mode:** Non-unified identity between object and file (Default mode)
- **unified_mode:** Unified identity between object and file

The mode is represented by the **id_mgmt** configuration parameter in the `object-server-sof.conf` file:

```
id_mgmt = local_mode | unified_mode
```

You can change this parameter by using the **mmobj config change** command. For more information, see “Configuring authentication and setting identity management modes for unified file and object access” on page 143.

Note:

- If you are upgrading from IBM Spectrum Scale 4.1.1, **id_mgmt = local_mode** is the default setting.
- Only one mode can be effective at a given time and it needs to be configured by the administrator for the entire system. **id_mgmt = local_mode** is the default setting.
- If you plan to use `unified_mode`, the authentication mechanism for file and object must be the same. If you set **id_mgmt** to `unified_mode` and the file authentication and object authentication are not common, then the ID resolution of the users will either not work or it will be incorrect. This will lead to either object not being created with 503 error* return code or object being created with improper user ID. Therefore, it is very important that the administrators ensure that a common authentication with appropriate ID mapping is configured for file and object.

* If you are using swift client, instead of 503, you might get an error similar to the following:

```
'put_object('container_name', 'object_name', ..) failure and no ability to reset contents for reupload.'
```

local_mode - non-unified identity between object and file

- Use-case for unified file and object access in `local_mode`:
 - Data created from the object interface is available for application to run analytics using the file interface, where ownership of files is not essential.

- Data created from the file interface is accessible from the object interface after objectization of those files.
- To address this use case, object authentication setup is independent of file authentication setup. Although, you can set up object and file authentication from a common authentication server in case of AD or LDAP.
- Objects created or updated using the object interface are owned by the swift user. Application processing the object data from file interface need the required file ACL to access the object data.
- Data updated from the file interface after objectization is available for object access.
- Containers created with a unified file and object access policy that are exposed as export points need appropriate ACLs set as needed by SMB, NFS, and POSIX.
- If the object already exists, existing ownership of the corresponding file is retained if `retain_owner` is set to `yes` in `object-server-sof.conf`. For more information, see “Configuration files for IBM Spectrum Scale for object storage” on page 153.
- Retaining ACL, extended attributes (xattrs), and Windows attributes (winattrs): If the object is created or updated over existing file then existing file ACL, xattrs, and winattrs are retained if `retain_acl`, `retain_xattr`, and `retain_winattr` are set to `yes` in `object-server-sof.conf`. For more information, see “Configuration files for IBM Spectrum Scale for object storage” on page 153.

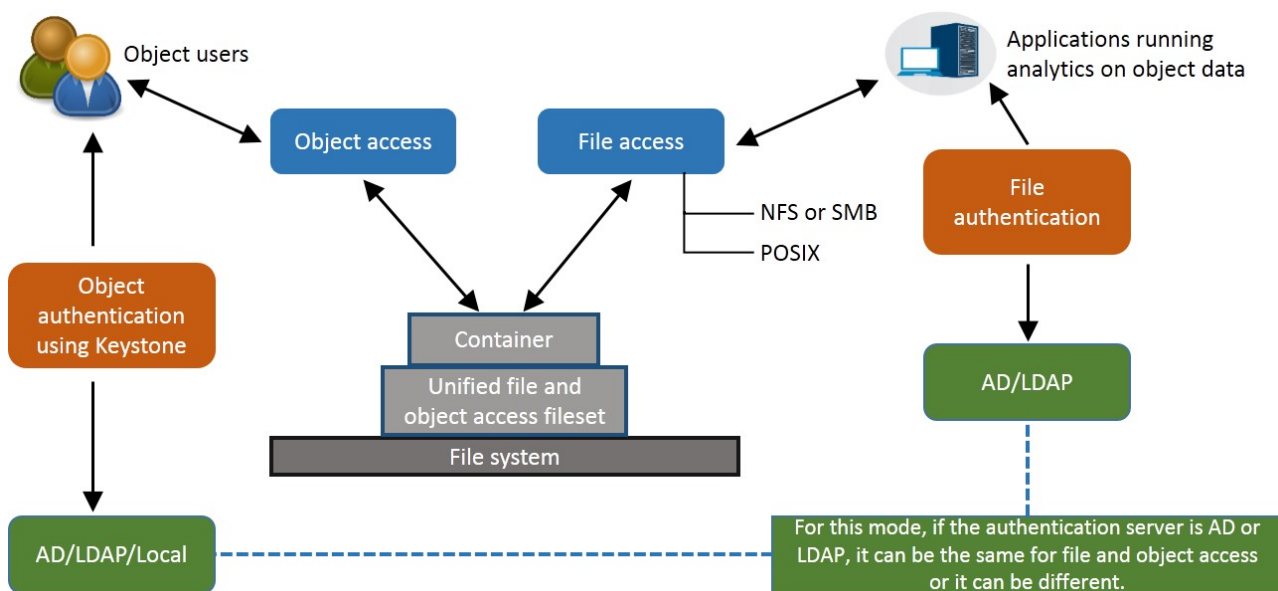


Figure 3. `local_mode` - non-unified identity between object and file

unified_mode - unified identity between object and file

- Users from object and file are expected to be common and coming from the same directory service (only AD+RFC 2307 or LDAP).

Note: If your deployment only uses SMB based file interface and none other for file access and file authentication is configured with Active Directory (AD) with Automatic ID mapping, unified file and object access can be used, assuming that object is configured with the same AD domain.

- Ownership: Object created from the object interface is owned by the user doing the object PUT operation.
- If the object already exists, existing ownership of the corresponding file is retained if `retain_owner` is set to `yes` in `object-server-sof.conf`. For more information, see “Configuration files for IBM Spectrum Scale for object storage” on page 153.

- Authorization: Object access follows the object ACL semantics and file access follows the file ACL semantics.
- Retaining ACL, extended attributes (xattrs), and Windows attributes (winattrs): If the object is created or updated over existing file then existing file ACL, xattrs, and winattrs are retained if `retain_acl`, `retain_xattr`, and `retain_winattr` are set to `yes` in `object-server-sof.conf`. For more information, see “Configuration files for IBM Spectrum Scale for object storage” on page 153.
- When a user does a PUT operation for an object over an existing object or does a PUT operation for a fresh object over a nested directory, no explicit file ACL is set for that user. This means that it is possible that in some cases, the user might not have access to that file from the file interface even though the user has access from the object interface. This is done to prevent changing of the file ACL from the object interface to maintain file ACL semantics. In such cases, if the user is required to have permission to access the file also, explicit file ACL permission need to be set from the file interface.
For example: If user Bob performs a PUT operation for an object over an existing object (object maps to a file) owned by user Alice, Alice continues to own the file and there is no explicit file level ACL that is set for Bob for that file. Similarly, when Bob performs a PUT operation for a new object inside a subdirectory (already created by Alice), no explicit file ACL is set on the directory hierarchy for Bob. Bob does not have access to the object from the file interface unless there is an appropriate directory inheritance ACL that is set. To summarize, the object ingest does not change any file ACL and vice versa.

Table 9. Object input behavior in `unified_mode`.

Note: In the scenarios listed in the following table, the operations are being done by user **Bob** from the object interface. The instances of owned by user Alice imply that the file or directory ownership maps to user **Alice** from the file side. Also, it is assumed that the `retain_owner`, `retain_acl`, `retain_xattr`, and `retain_winattr` parameters are set to `yes` in `object-server-sof.conf`.

Operation from SWIFT interface on object or container	Ownership result on corresponding file or directory		ACL, xattr, and winattr retention behavior on corresponding file or directory	
	File	Directory	File	Directory
Bob does a PUT operation for an object that is not present	The ownership of the file is set to Bob	NA	Default GPFS ACLs are set	NA
Bob does a PUT operation for a container that is not present	NA	The ownership of the directory is set to Bob	NA	Default GPFS ACLs are set
Bob does a PUT operation for an object that is already present and is owned by Alice	The ownership of the file continues to be with Alice. Bob is not given any file ACL explicitly.	No changes in the ownership of the parent directory	Existing ACL, file xattrs, and file winattrs are retained**	NA
Bob does a POST operation (update metadata) of existing object owned by Alice	The ownership of the file continues to be with Alice. Bob is not given any file ACL explicitly	NA	Existing ACL, file xattrs, and file winattrs are retained**	NA
Bob does a POST operation (update metadata) of existing container owned by Alice	NA	The ownership of the directory continues to be with Alice. Bob is not given any directory ACL	NA	NA

Table 9. Object input behavior in *unified_mode* (continued).

Note: In the scenarios listed in the following table, the operations are being done by user **Bob** from the object interface. The instances of owned by user Alice imply that the file or directory ownership maps to user **Alice** from the file side. Also, it is assumed that the `retain_owner`, `retain_acl`, `retain_xattr`, and `retain_winattr` parameters are set to `yes` in `object-server-sof.conf`.

Operation from SWIFT interface on object or container	Ownership result on corresponding file or directory		ACL, xattr, and winattr retention behavior on corresponding file or directory	
Bob does a POST operation (update ACL) of existing container owned by Alice	NA	The ownership of the directory continues to be with Alice. Bob is not given any directory ACL	NA	NA
GET/DELETE/HEAD	No impact			

Note: **Unified file and object access retains the extended attributes (xattr), Windows attributes (winattrs) and ACL of the file if there is a PUT request from an object over an existing file. However, security or system namespace of extended attributes and other IBM Spectrum Scale extended attributes such as immutability, pcache, etc. are not retained. Swift metadata (`user.swift.metadata`) is also not retained and it is replaced according to object semantics which is the expected behavior.

Advantages of using `unified_mode`

IBM Spectrum Scale offers various features that leverage user identity (UIDs or GIDs). With `unified_mode`, you can use these features seamlessly across file and object interfaces.

- **Unified access to object data:** User can access object data using NFS or SMB exports using their AD or LDAP credentials.
- **Quota:** Quota for users that work on UID or GID can be set such that they work for the file as well as object interface.

Example: User A can have X quota on a unified access fileset assigned using GPFS quota commands which can hold true for all data created by the user from the file or the object interface.

For more information, see the *Quota related considerations for unified_mode* section.

- **ILM:** Tiering of user specific data leveraging UID or GID.

Example 1: The UID and GID file attributes can be used to create an ILM placement policy to place the files owned by the Gold customers in faster storage pools and retain the files in the pools even when the pool storage starts reaching the threshold. The UID and GID file attributes can also be used to create a migration ILM policy so that, when the pool reaches its storage threshold, all files older than 30 days are moved to a slower storage pool except the ones owned by the Gold customers.

Example 2: After a user has left the organization, the UID of the user can be used to migrate the data and retain it on the archive tape for as long as defined as defined by the ILM policies.

- **Backup:** Backup of user specific data leveraging UID or GID.

Example: UID and GID file attributes in the policy rules that are defined for the `mmbackup` command can be used to regularly back up the data of selective users.

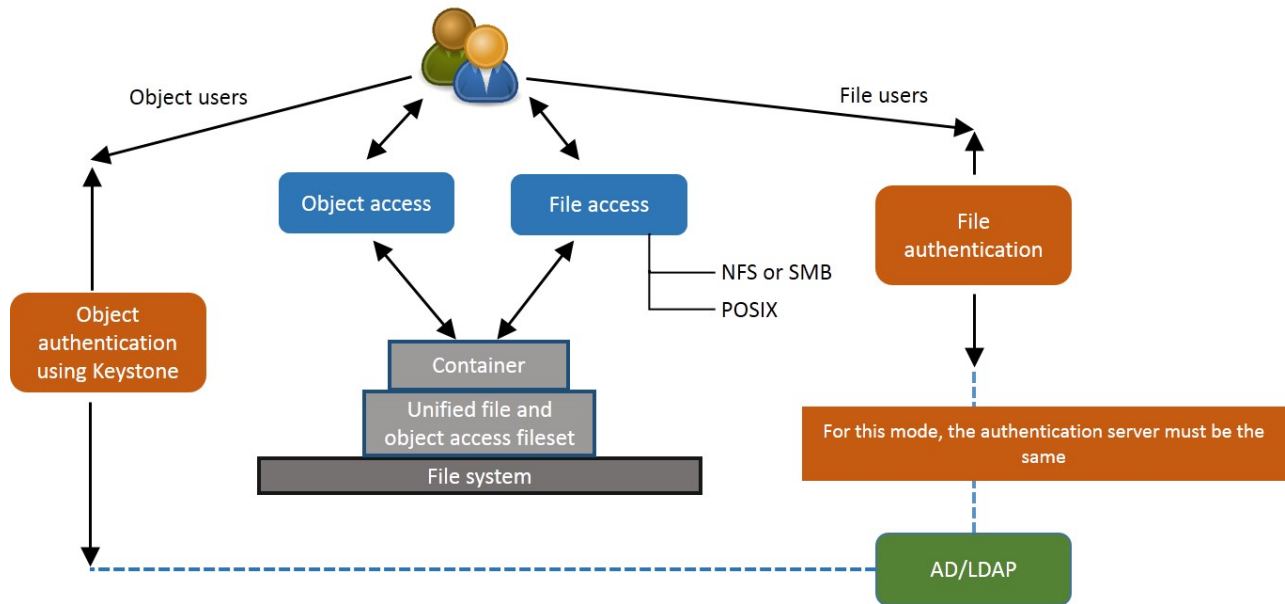


Figure 4. *unified_mode* - unified identity between object and file

Quota related considerations for *unified_mode*

There are three types of quotas that need to be considered:

- Quota for a user set using file system commands for that fileset which is set using User ID or Group ID. This quota represents the size in bytes up to which the user can create data on a given fileset. This is tracked at the file system level.
- Container quota: This is the size in bytes of objects that can be stored in a container with no mapping with the user. For more information, see OpenStack documentation of container quotas.
- Account quotas: See OpenStack documentation of account quotas.

The fileset quotas and container level quotas as well as fileset quotas and account quotas are independent of each other.

In some cases, the fileset quota should be cumulative of all the containers' quotas hosted over it, though it is not mandatory. When both the quotas at the fileset level as well as at the container quota level are set, and if the fileset quota is reached, no more object data can be input on any of the containers hosted by that fileset, despite of the container quota not being reached. Hence, when you plan to use both the quotas, it is important to understand these details.

The objectization process does not take into account the container quota and the account quota. This means that there might be a scenario where a container can host more data than the container quota associated with it especially when the **ibmobjectizer** service has objectized files as objects.

For example, consider that:

- You want to have a total of 1 TB of data allocated for file and object access.
- You want each user to have an overall quota from the file as well as the object interface to be 10 GB.
- You have a pre-defined set of 100 containers which are enabled for object and file access (using the storage policy for object storage) and users access to different containers is dependent on the container ACLs.

In this case, quotas are set as follows:

1. Set the fileset quota associated with the file access policy to 1 TB.

2. Set the user quota on that fileset to 10 GB.
3. Set the container quota to the required level. However, setting it more than fileset quota cannot be honored until fileset quota is increased or unset.

In this example scenario, note that the object access will be restricted if either the user quota or the fileset quota is reached, even though the container quota is not reached.

Authentication in unified file and object access

The following section gives information about how file authentication and object authentication are configured for different identity management modes.

local_mode - non-unified identity between object and file

This is a non-unified ID mode. In this mode, all the objects created continue to be owned by the `swift` user, that is a special user under whose context the object server runs on the system. Because in this mode there is no ID mapping of objects to user ID, object authentication can be configured to any supported authentication schemes and file authentication can continue to be configured to any supported authentication scheme.

For supported authentication schemes, see the *Authentication support matrix* table in the *Authentication considerations* topic in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

unified_mode - unified identity between object and file

This mode allows objects and files to be owned by the users' UID and the corresponding GID that created them. This mode mandatorily requires both the object protocol and the file protocol to be configured with the same authentication scheme. The supported authentication schemes for the unified mode are:

- AD for Authentication + RFC 2307 for ID mapping
- LDAP for authentication as well as for ID mapping

Note: User-defined authentication is not supported with both the identity management modes.

The objectizer process

The objectization process converts files ingested from the file interface on unified file and access enabled container path to be available from the object interface. The name of the service that does this is **ibmobjectizer**.

When new files are added from the file interface, they need to be visible to the Swift database to show correct container listing and container or account statistics.

The **ibmobjectizer** service ensures synchronization between the file metadata and the object metadata at predefined time interval that ensures accurate container and account listing. The **ibmobjectizer** service identifies new files added from the file interface and adds the Swift system metadata to them so that they are objectized. The **ibmobjectizer** service then determines its container and account databases and adds a new object entry to those. It also identifies files deleted from file interface and deletes their corresponding entries from container and account databases.

This is particularly useful in setups where data is ingested using legacy file interface based devices such as medical and scientific devices and it needs to be stored and accessed over cloud using the object interface.

The **ibmobjectizer** service is a singleton and it is started when object is enabled and the file-access object capability is set. However, the **ibmobjectizer** service starts objectization only when there are containers with unified file and object access storage policies configured and the file-access object capability is set.

To identify the node on which the **ibmobjectizer** service is running, use the **mmces service list --verbose** command.

Attention: If object services on the singleton node are stopped by the administrator manually, objectization is stopped across the cluster. Therefore, manually stopping services on a singleton node must be planned carefully after understanding its impact.

For information on limitations on the objectizer process, see “Limitations of unified file and object access” on page 150.

Related concepts:

“Understanding and managing Object services” on page 123

Use the following information to manage services related to IBM Spectrum Scale for object storage.

Related tasks:

“Setting up the objectizer service interval” on page 143

Take the following steps to set up the objectizer service interval.

Related reference:

“Configuration files for IBM Spectrum Scale for object storage” on page 153

Use the following information to manage options in configuration files that are used for IBM Spectrum Scale for object storage including the unified file and object access feature. These configuration files are located in the `/etc/swift` directory.

File path in unified file and object access

One of the key advantages of unified file and object access is the placement and naming of objects when stored on the file system.

Unified file and object access stores objects following the same path hierarchy as the object's URL. In contrast, the default object implementation stores the object following the mapping given by the ring, and its final file path cannot be determined by the user easily. For example, an object with the following URL is stored by the two systems as follows:

- **Example object URL:** `https://swift.example.com/v1/acct/cont/obj`
- **Path in default object implementation:** `/ibm/gpfs0/object_fileset/o/z1device108/objects/7551/125/75fc66179f12dc513580a239e92c3125/75fc66179f12dc513580a239e92c3125.data`
- **Path in unified file and object access:** `/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_763476384728498323747/cont/obj`

In this example, it is assumed that the object is configured over the `/ibm/gpfs0` file system with the default object located on the `object_fileset` fileset and the unified file and object access data is located under the `obj_sofpolicy1` fileset. `s69931509221z1device1` is auto-generated based on the swift ring parameters and `AUTH_763476384728498323747` is auto-generated based on the account ID from keystone.

Attention: Do not unlink object filesets including the unified file and object access enabled filesets.

Determining the POSIX path of a unified file and object access enabled fileset

Use the following steps for determining the POSIX path of a unified file and object access enabled fileset.

1. List all storage policies for object.

```
mmobj policy list
```

Index	Name	Default	Deprecated	Fileset	Functions
0	SwiftDefault	yes		object_fileset	

```

13031510160 sof-policy1                obj_sof-policy1    file-and-object-access
13031510260 CompressionTest            yes                obj_CompressionTest compression
13031510290 CompressionDebug           yes                obj_CompressionDebug compression
13031511020 CompressionNew              obj_CompressionNew compression

```

- Note the index and fileset name for the policy you are interested in, and using the **mm1sfileset** command determine the junction point.

```

mm1sfileset fs0 | grep obj_sof-policy1
obj_sof-policy1      Linked    /ibm/fs0/obj_sof-policy1

```

The swift ring builder creates a single virtual device for unified file and object access policies, named with the region number, starting with **s** and appended with **z1device1**. For example:

```
s13031510160z1device1
```

- List the swift projects and identify the one you are interested in working with.

```

openstack project list
+-----+-----+
| ID                               | Name   |
+-----+-----+
| 73282e8bca894819a3cf19017848ce6b | admin  |
| 1f78f58572f746c39247a27c1e0e1488 | service|
+-----+-----+

```

- Construct the account name by appending the project id with **AUTH_** (or substitute the correct project prefix if you've customized this). For the **admin** project, use:

```
AUTH_73282e8bca894819a3cf19017848ce6b
```

The full path to the unified file and object access containers is the concatenation of the fileset linkage, the virtual device name, and the account name:

```
/ibm/fs0/obj_sof-policy1/s13031510160z1device1/AUTH_73282e8bca894819a3cf19017848ce6b/
```

- List the containers defined for this account.

```

ls /ibm/fs0/obj_sof-policy1/s13031510160z1device1/AUTH_73282e8bca894819a3cf19017848ce6b/
new1    fifthcontainer  RTC73189_1  RTC73189_3  RTC73189_5  RTC73189_7  sixthcontainer

```

The following is an example of determining the POSIX path of a unified file and object access enabled fileset from the command line with one rather complex command. This assumes that your GPFS file system device is **gpfs0**, and also that you are interested in the first storage policy listed. If this is not the case, update the script accordingly.

Commands used:

```

# openstack project list
# swift capabilities

```

Assumptions:

```

# File system name is is gpfs0
# Project/Account name is admin
# first [1] SoF policy is considered

```

```

echo $(find $(mm1sfileset gpfs0 | grep $(perl -e '$p=~swift capabilities
| grep policies:~;$p=~s/./=>/g;eval"\$v= {\".$p.\"}";print$v->{policies}[1]{name};')
| awk '/ / {print $3}') -name AUTH_$(openstack project list | awk '/ admin / {print $2}'))

```

Administering unified file and object access

Use the following information to administer unified file and object access in your IBM Spectrum Scale setup.

Enabling the file-access object capability

Before you can use unified file and object access, you must enable the file-access object capability on the whole cluster.

- Enable the file-access object capability using the **mmobj config change** as follows.

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities --property file-access-enabled --value true
```

- Verify that the file-access object capability is enabled using the **mmobj config list** as follows.

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities --property file-access-enabled
```

The system displays output similar to the following:

```
file-access-enabled = true
```

Setting up the objectizer service interval

Take the following steps to set up the objectizer service interval.

The default interval between the completion of an objectizer cycle and the starting of the next cycle is 30 minutes. However, this needs to be planned properly based on the following:

- The frequency and the number of new file ingestions that you are expecting to be objectized.
- The number of protocol nodes you have deployed.
- How quickly you need the ingested file to be objectized.

Note: Objectization is a resource intensive process. The resource utilization is related to the number of containers that have unified file and object access enabled. The schedule of running the objectization process must be planned carefully. Running it too frequently might impact your protocol node's resource utilization. It is recommended to either schedule it during off business hours, especially if you have a small number of protocol nodes (say 2) with basic resource configuration, or schedule with an interval of 30 minutes or more if you have protocol nodes with adequate resources (where the number of protocol nodes > 2). **It is recommended to set the objectizer service interval to a minimum of 30 minutes or more irrespective of your setup.** If you need to urgently objectize files then you can use the **mmobj file-access** command that allows you to immediately objectize the specified files.

- Set up the objectization interval using the **mmobj config change** as follows.

```
mmobj config change --ccrfile spectrum-scale-objectizer.conf \  
--section DEFAULT --property objectization_interval --value 2400
```

This command sets an interval of 40 minutes between the completion of an objectization cycle and the start of the next cycle.

- Verify that the objectization time interval is changed using the **mmobj config list** as follows.

```
mmobj config list --ccrfile spectrum-scale-objectizer.conf --section DEFAULT  
--property objectization_interval
```

Configuring authentication and setting identity management modes for unified file and object access

You can configure authentication and set the identity management modes for unified file and object access using the following steps.

The identity management modes for unified file and object access are set in the `object-server-sof.conf` file. The default mode is `local_mode`.

Note: It is important to understand the identity management modes for unified file and object access and set the mode you want accordingly. Although it is possible to move from one mode to another, some considerations apply in that scenario.

The `unified_mode` identity management mode for unified file and object access is supported only with Active Directory (AD) with UNIX-mapped domains and LDAP authentication configurations. This mode must not be configured with local or user-defined authentication configurations.

Important: If you are using `unified_mode`, the authentication for both file and object access must be configured and the authentication schemes must be the same and configured with the same server. If not, the request to create object might fail with user not found error.

Use the following steps on a protocol node to configure authentication and enable `unified_mode`.

1. Determine which authentication scheme best suits your requirements. You can use either LDAP or AD with UNIX-mapped domains.

Note: Because object can be configured with only one AD domain, you need to plan which of the UNIX-mapped AD domains, in case there are trusted domains, is to be configured for object.

2. Configure file access using the **mmuserauth** command as follows.

```
mmuserauth service create --data-access-method file
--type ad --servers myADserver --idmap-role master
--netbios-name scale --unixmap-domains 'DOMAIN(5000-20000)'
```

3. Configure object access using the **mmuserauth** command as follows.

```
mmuserauth service create --data-access-method object --type ad
--user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
--base-dn "dc=IBM,DC=local" --ks-dns-name c40bbc2xn3 --ks-admin-user admin
--servers myADserver --user-id-attrib cn --user-name-attrib sAMAccountName
--user-objectclass organizationalPerson --user-dn "cn=Users,dc=IBM,dc=local"
--ks-swift-user swift --ks-swift-pwd Passw0rd
```

4. Change **id_mgmt** in the object-server-sof.conf file using the **mmobj config change** command as follows.

```
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property id_mgmt --value unified_mode
```

5. If object authentication is configured with AD, set **ad_domain** in the object-server-sof.conf file.

```
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property ad_domain --value POLLUX
```

Note: Do not specify **ad_domain** with LDAP configurations.

To find the correct **ad_domain** name, use the following command:

```
/usr/lpp/mmfs/bin/net ads lookup -S {AD_SERVER_NAME | AD_SERVER_IP} -d0
```

For example, in the output of the following command, the value of the **Pre-Win2k Domain** field is the **ad_domain**.

```
/usr/lpp/mmfs/bin/net ads lookup -S 192.196.79.34 -d0
```

```
...
Forest: pollux.com
Domain: pollux.com
Domain Controller: win2k8.pollux.com
Pre-Win2k Domain: POLLUX
Pre-Win2k Hostname: WIN2K8
Server Site Name : Default-First-Site-Name
Client Site Name : Default-First-Site-Name
...
```

Your unified file and object access enabled fileset is now configured with **unified_mode**.

6. List the currently configured **id_mgmt** mode using the **mmobj config list** command as follows.

```
mmobj config list --ccrfile object-server-sof.conf --section DEFAULT --property id_mgmt
```

Important:

1. If the PUT requests fail in **unified_mode**, check if the user name is resolvable on the protocol nodes using the following command:

```
id '<user_name>'
```

If user name in AD is in the **domain\user_name** format, use the following command:

```
id '<domain>\<user_name>'
```

2. Ensure that the **ad_domain** parameter is not present in the object-server-sof.conf file when LDAP is configured.

- To list the object-server-sof.conf file contents, use the following command:

```
mmobj config list --ccrfile object-server-sof.conf
```

- If **ad_domain** is present, remove it as follows:
 - a. Copy `/etc/swift/object-server-sof.conf` to a temporary location, say `/tmp`.
 - b. Modify the temporary file by appending a '-' before the **ad_domain** parameter. This marks that parameter for deletion.
 - c. Upload the modified file using the following command:


```
mmobj config change --ccrfile object-server-sof.conf --merge-file /tmp/object-server-sof.conf
```
 - d. **[Optional]:** Validate that **ad_domain** is removed from the `object-server-sof.conf` file by listing the file contents.
- 3. Configuring file authentication with the same scheme as that of object authentication is a mandatory prerequisite before you enable the **unified_mode** identity management mode. In case you configure file authentication later, you must restart swift on the file server for the changes to be effective. You can do this by changing **id_mgmt** to `local_mode` and then changing it back to `unified_mode` using the following commands.

```
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property id_mgmt --value local_mode
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property id_mgmt --value unified_mode
```

Creating or using a unified file and object access storage policy

Use the following steps to create or use a unified file and object access storage policy.

1. Create a unified file and object access storage policy using the **mmobj policy create** command. This step also creates a fileset.

For example:

```
mmobj policy create sof-policy1 --enable-file-access
```

The system displays output similar to the following:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_sof-policy1
[I] Creating new unique index and build the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

2. List the available storage policies using the **mmobj policy list** command and determine which policies are for unified file and object access by viewing the **Functions** column of the output.

For example:

```
mmobj policy list --verbose
```

The system displays output similar to the following:

Index	Name	Deprecated	Fileset	Fileset Path	Functions	Function Details
0	SwiftDefault		object_fileset	/ibm/cesSharedRoot/object_fileset		
11751509160	sof-policy1		obj_sof-policy1	/ibm/cesSharedRoot/obj_sof-policy1	file-and-object-access	regions="1"
11751509230	mysofpolicy		obj_mysofpolicy	/ibm/cesSharedRoot/obj_mysofpolicy	file-and-object-access	regions="1"
11751510260	Test19		obj_Test19	/ibm/cesSharedRoot/obj_Test19		regions="1"

3. Start using one of these storage policies to create data in a unified file and object access environment.

For more information, see the following:

- “Associating containers with unified file and object access storage policy”
- “Creating exports on container associated with unified file and object access storage policy” on page 146

For information on mapping of storage policy and filesets, see “Mapping of storage policies to filesets” on page 130.

You must create export at the container level. From NFS or SMB, if you create a peer container, base containers created from NFS and SMB cannot be multiprotocol.

Associating containers with unified file and object access storage policy

Use the following steps to associate a container with a unified file and object access storage policy.

1. Associate a container with a unified file and object access storage policy using the following command.

```
swift post container1 --os-auth-url http://specscaleswift.example.com:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
--header "X-Storage-Policy: sof-policy1"
```

In this **swift post** example, the storage policy is specified with the customized header X-Storage-Policy using the **--header** option.

2. Upload an object in the container associated with the unified file and object access storage policy using the following command.

```
swift upload container1 imageA.JPG --os-auth-url http://specscaleswift.example.com:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
object1
```

Note: The steps performed using **swift** commands can also be done using **curl**. For more information, see “curl commands for unified file and object access related user tasks” on page 152.

Creating exports on container associated with unified file and object access storage policy

Use the following steps to create an NFS or SMB export on the directory that maps to the container associated with the unified file and object access storage policy.

Create an NFS or SMB export on the directory that maps to the container associated with the unified file and object access storage policy.

1. Create the NFS export as follows:

```
mmnfs export add "/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/
AUTH_763476384728498323747/cont"
```

2. Create the SMB export as follows:

```
mm smb export add smbexport "/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/
AUTH_763476384728498323747/cont"
```

Note:

- It is strongly recommended that you create file exports on or below the container path level and not above it. Creating file exports above the container path level might lead to deletion of the unified file and object access enabled containers which is undesirable.
- When using POSIX interface, it is strongly recommended to only allow access of data to POSIX users from on or below the container path. Accidental deletion of container or data above might lead to inconsistent state of the system.

Enabling object access for selected files

Use the following steps to objectize files under all the containers associated with the unified file and object access storage policy under an account

In a unified file and object access environment, if you want to access files created from file interfaces such as POSIX, NFS, or CIFS through object interfaces such as curl or swift, you need to make these files available for the object interface. For making these files available for the object interface, the **ibmobjectizer** service, once activated, runs periodically and makes newly created files available for the object interface. You can also use the **mmobj file-access** command to selectively enable files for access through the object interface.

The purpose of this command is to make certain files available to object sooner (or immediately) than when the objectizer would have made them available. This command does not ensure synchronization between file and object data. Therefore, files deleted are not immediately reflected in the object interface. Complete synchronization is done by the **ibmobjectizer** service eventually.

In unified file and object access enabled filesets, files can be accessed from the object interface if you know the entire URI (including keystone account ID, device etc.) to access that file without the need for them to be objectized either using the **ibmobjectizer** service or the **mmobj file-access** command.

Note: The **mmobj file-access** command does not enable or disable the unified file and object access feature. It is only used to objectize files (that is enable object access for files) immediately when initiated by the administrator. Disabling object access for files is not supported.

- To objectize files under all the containers associated with the unified file and object access storage policy under an account, use the **mmobj file-access** command as follows:

```
mmobj file-access --storage-policy sof_policy --account-name admin
```

The system displays output similar to the following:

```
Loading objectization configuration from CCR
Fetching storage policy details
Creating container to database map
Performing objectization
Objectization complete
```

This command objectizes all containers in the account admin and enables them for access through the object interface.

- To objectize files under a container, use the **mmobj file-access** command as follows:

```
mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1
```

This command objectizes all files in container1 and enables them for access through the object interface.

- To objectize a file while specifying a storage policy, use the **mmobj file-access** command with the **--storage-policy** option as follows:

```
mmobj file-access --storage-policy sof_policy --account-name admin \
--container-name container1 --object-name file1.txt
```

This command objectizes file1.txt in container1 and enables it for access through the object interface.

- To objectize a file, use the **mmobj file-access** command as follows:

```
mmobj file-access --object-path \
/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_763476384728498323747/cont/file1.txt
```

This command objectizes file1.txt at location /ibm/cesSharedRoot/fileset1/Auth_12345/container1/ and enables it for access through the object interface.

For more information about the **mmobj file-access** command, see the “mmobj command” on page 581.

Example scenario - administering unified file and object access

The following example describes an end-to-end scenario of administering and using unified file and object access.

Before you can use the following steps, IBM Spectrum Scale for object storage must be installed.

This example provides a quick reference of steps performed for unified file and object access. For detailed information about these steps, see “Administering unified file and object access” on page 142.

1. Enable the file-access object capability as follows.

```
mmobj config change --ccrfile spectrum-scale-object.conf \
--section capabilities --property file-access-enabled --value true
```

2. [Optional] Change the objectizer service interval as follows.

```
mmobj config change --ccrfile spectrum-scale-objectizer.conf \
--section DEFAULT --property objectization_interval --value 600
```

3. [Optional] Change the identity management mode to unified_mode as follows.

```
mmobj config change --ccrfile object-server-sof.conf \
--section DEFAULT --property id_mgmt --value unified_mode
```

4. [Optional] Set the ad_domain parameter as follows.

```
mmobj config change --ccrfile object-server-sof.conf \
--section DEFAULT --property ad_domain --value ADDOMAINX
```

5. Create a unified file and object access storage policy as follows.

```
mmobj policy create SwiftOnFileFS --enable-file-access
```

The system displays output similar to the following:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_SwiftOnFileFS
[I] Creating new unique index and building the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

This command also creates a unified file and object access enabled fileset.

6. Create a base container with a unified file and object access storage policy as follows.

```
swift post unified_access -H "X-Storage-Policy: SwiftOnFileFS"
```

7. Store the path created for the container by finding it in the newly created fileset as follows.

```
export FILE_EXPORT_PATH=`find /ibm/gpfs0/obj_SwiftOnFileFS/
-name "unified_access" `
```

```
# echo $FILE_EXPORT_PATH
/ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access
```

8. Create an SMB export on the path as follows.

```
mmsmb export add unified_access $FILE_EXPORT_PATH
```

The system displays output similar to the following:

```
mmsmb export add: The SMB export was created successfully
```

9. Create an NFS export on the path.

```
mmnfs export add $FILE_EXPORT_PATH --client
"*(Access_Type=RW,Squash=no_root_squash,SecType=sys)"
```

The system displays output similar to the following:

```
192.0.2.2: Redirecting to /bin/systemctl stop nfs-ganesha.service
192.0.2.3: Redirecting to /bin/systemctl stop nfs-ganesha.service
192.0.2.2: Redirecting to /bin/systemctl start nfs-ganesha.service
192.0.2.3: Redirecting to /bin/systemctl start nfs-ganesha.service
NFS Configuration successfully changed. NFS server restarted on all NFS nodes.
```

10. Check the NFS and SMB exports.

```
mmnfs export list
Path                                     Delegations Clients
-----
/ibm/gpfs0/obj_SwiftOnFileFS/
s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access  none      *
```

```
mmsmb export list
export      path                                     guest ok   smb encrypt
unified_access /ibm/gpfs0/obj_SwiftOnFileFS/
s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access  no        auto
```

Information:
The following options are not displayed because they do not contain a value:
"browseable"

11. Access this export with NFS or SMB clients and create a sample directory and a file. For example: DirCreatedFromGPFS/File1.txt and DirCreatedFromSMB/File2.txt

You can view the association of ownership when data is created from the SMB interface as follows.

```
ls -l /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB
total 0
-rwxr--r--. 1 ADDOMAINX\administrator
ADDOMAINX\domain users 20 Oct 21 18:09 File2.txt
```

```
mmgetacl /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB
```

```

#NFSv4 ACL
#owner:ADDDOMAINX\administrator
#group:ADDDOMAINX\domain users
special:owner@:rwx:allow
(X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (X)SYNCHRONIZE
(X)READ_ACL (X)READ_ATTR (X)READ_NAMED
(-)DELETE (X)DELETE_CHILD (X)CHOWN
(X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (X)WRITE_NAMED

special:group@:r-x:allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE
(X)READ_ACL (X)READ_ATTR (X)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN
(X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

special:everyone@:r-x:allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE
(X)READ_ACL (X)READ_ATTR (X)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN
(X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

```

You can view the container and the file created from the REST interface and retention of ownership in the PUT operation as follows.

```

ls -l /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB/File2.txt

-rwxr-xr-x. 1 ADDDOMAINX\administrator ADDDOMAINX\domain users 520038360 Nov 3 11:47
/ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/AUTH_09271462d54b472c82adecff17217586/
DirCreatedFromSMB/unified_access/File2.txt

```

- Objectize that file immediately by using the following command or wait for the objectization cycle to complete.

```

mmobj file-access --object-path \
/ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/AUTH_09271462d54b472c82adecff17217586/
unified_access/File2.txt

```

- Download that object using the Swift client which is configured with all variables as follows.

```
swift download unified_access/File2.txt
```

- List the contents of the container using the Swift client which is configured with all variables as follows.

```
swift list unified_access
```

The system displays output similar to the following:

```
DirCreatedFromGPFS/File1.txt
DirCreatedFromSMB/File2.txt
```

Note: The steps performed using **swift** commands can also be done using **curl**. For more information, see “curl commands for unified file and object access related user tasks” on page 152.

In-place analytics using unified file and object access

Use the following information to leverage in-place object data analytics using unified file and object access.

Unified file and object access is one of the key features of IBM Spectrum Scale for object storage that enables direct object access as files from the traditional file access such as POSIX, NFS or SMB and vice versa. Using object storage policies for containers you can have object ingested in IBM Spectrum Scale for object storage be accessed as files as well as allow files ingested using file protocols available for object access. This feature enables data analytics of object data hosted on IBM Spectrum Scale, where you can leverage in-place object data analytics. IBM Spectrum Scale supports Hadoop connectors using which you can run analytics on the object data which is accessible from the file interface and generates in-place results which are directly accessible from the object interface. This prevents any movement of data across object interfaces and thus proves to be a suitable platform for object storage as well as integrated in-place analytics for the data hosted by it.

The following diagram shows an IBM Spectrum Scale object store with unified file and object access. The object data is available as file on the same fileset. IBM Spectrum Scale Hadoop connectors allow the data to be directly leveraged for analytics.

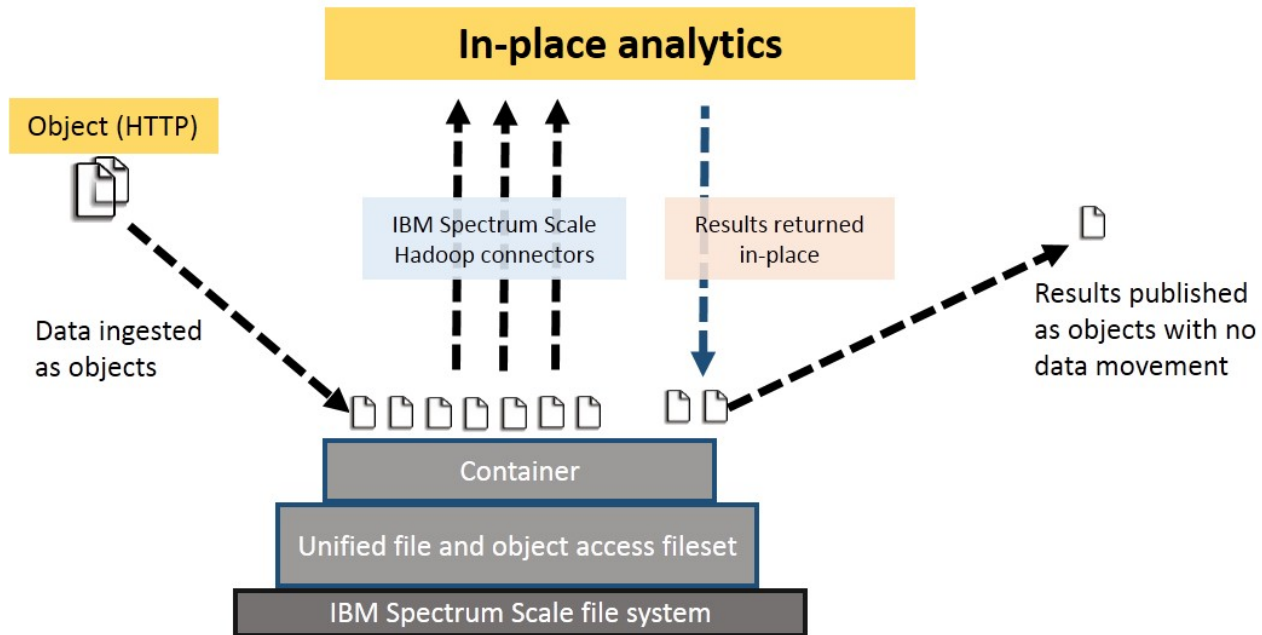


Figure 5. In-place analytics with unified file and object access

Limitations of unified file and object access

The following limitations apply for unified file and object access in IBM Spectrum Scale.

- Existing file data cannot be enabled for object access. The base container must be created from the object interface in the fileset being used for the unified access storage policy and then only the data added after that is enabled for object access.
- Concurrent access to the same object or file from file and object interface at the same time will lead to an undefined state. There are a variety of ways to prevent conflicts. For example:
 1. You can have your workflow enforce this.
 2. You can explicitly enforce read-only access for some periods. With NFS and SMB, it can be done in the export definition. With POSIX, it can be done using ACLs.
- Files or directories created at the base container level cannot be enabled for object access. Only the files created under the container are enabled for object access.
- Multi-region object deployment cannot be used with unified file and object access.
- Object versioning is not supported with unified file and object access.
- Special files such as device files and pipes, and soft links can exist in the object container directory, but they are not visible from the object interface.
- AFM-based Async DR is supported with unified file and object access. No other active file management (AFM) modes are supported with unified file and object access.
- Containers must be deleted from the object interface. Container directories deleted from the file interface continue to show up in the container listing, until the container is deleted from the object interface.
- GPFS quota and Swift container and account quota are mutually exclusive in Release 4.2 and later. The user quota assigned to a user or a group in GPFS does not relate to the container quota defined in the object interface.

- Swift large object support (dynamic large object and static large object) is not available with unified file and object-enabled containers. S3 multipart uploads are also not supported with unified file and object-enabled containers.
- GPFS immutability is not supported with unified file and object access.
- Only object metadata can be viewed and modified from the object interface. Extended attributes defined from the file interface cannot be viewed from the object interface.
- Empty directories created from the file interface within a container are not objectized and they are not listed in the container listing.
- Files or directories with ":" or newline characters in their names are not supported and these files or data residing in these containers are not objectized.
- Change of authentication scheme of file or object could directly impact access to existing file or object data. Therefore, change of authentication is not supported as it results in loss of access for the users to the existing data on the system.
- Object ETag is inaccurate in the following scenarios:
 - Whenever an object is modified from the file interface. In this case, performing a conditional request using 'If-Match' or 'If-None-Match' headers returns incorrect results.
 - ETag for files on an explicit GET request when the size of the object has changed (increased or decreased).
 - ETag for files on an explicit GET request when the content of the object has changed but the size has remained the same.
 - If the `user.swift.metadata` extended attribute is explicitly deleted from the file interface, ETag is not present because of which the headers do not return correct results. Users need to wait for at least one cycle of objectization or they need to explicitly objectize that file to use the ETag conditional request feature.

Constraints applicable to unified file and object access

The following constraints are applicable while creating and accessing objects and containers for unified file and object access:

Table 10. Configuration options for [swift-constraints] in swift.conf

Option	Limit
MAX_FILE_SIZE	5497558138880 (5 TiB)
MAX_META_NAME_LENGTH	128
MAX_META_VALUE_LENGTH	256
MAX_META_COUNT	90
MAX_META_OVERALL_SIZE	4096
MAX_HEADER_SIZE	8192
CONTAINER_LISTING_LIMIT	10000
ACCOUNT_LISTING_LIMIT	10000
MAX_ACCOUNT_NAME_LENGTH	256
VALID_API_VERSIONS	["v1", "v1.0"]
EXTRA_HEADER_COUNT	0

Data ingestion examples

Use the following example steps for data ingestion in the following scenarios.

- Data ingestion through object interface and access through file interface
- Data ingestion through file interface and access through object interface

- Data ingestion and access through object and file interfaces concurrently
1. **Standard REST client step:** Get proper authentication token from the Authentication URL using proper credentials to authorize on further requests.
 2. **Standard REST client step:** Using token obtained in the previous step perform PUT, POST, DELETE, COPY (object only), HEAD operations for objects under container created with unified file and object access storage policy.
 3. **Standard file client step:** Mount SMB or NFS exports on respective NFS or SMB clients with regular mount commands or interface available with file clients. For example:

```
mount -t cifs -o username=STORAGE5TEST\\fileuser1,password=Passw0rd5,vers=3.0 //192.0.2.4/unified_access /mnt/unified_access
```

Data ingestion using curl

In the following data ingestion example steps performed using **curl**, this setup is assumed:

- User: "fileuser"
- Password: "Password6"
- Account: "admin"
- Host: specscaleswift.example.com

1. Obtain the auth token using the following command:

```
curl -s -i -H "Content-Type: application/json"
-d '{ "auth": { "identity": { "methods": ["password"], "password": { "user": { "name": "fileuser", "domain":
{ "name": "Default" }, "password": "Passw0rd6" } } }, "scope": { "project": { "name": "admin", "domain": { "name": "Default" } } } } }'
http://specscaleswift.example.com:35357/v3/auth/tokens
```

The auth token obtained in this step is stored in the `$AUTH_TOKEN` variable.

2. Obtain the project list using the following command:

```
curl -s -H "X-Auth-Token: $AUTH_TOKEN" http://specscaleswift.example.com:35357/v3/projects
```

The project ID obtained in this step is stored in the `$AUTH_ID` variable.

3. Perform a PUT operation using the following command:

```
curl -i -s -X PUT --data @/tmp/file.txt -H "X-Auth-Token:
$AUTH_TOKEN" http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt
```

This command uploads the `/tmp/file.txt` file.

4. Set up the metadata age of the uploaded object using the following command:

```
curl -i -s -X POST -H "X-Auth-Token: $AUTH_TOKEN" -H
X-Container-Meta-Age:21 http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt
```

5. Read the metadata using the following command:

```
curl -i -s --head -H "X-Auth-Token: $AUTH_TOKEN"
http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt
```

curl commands for unified file and object access related user tasks

The following are the curl commands for performing user tasks related to unified file and object access.

For the following commands, it is assumed that:

- A token is generated and it is exported as an environment variable `auth_token`.
 - A swift endpoint URL for the project (tenant) for which token has been generated.
 - A unified file and object access storage policy named `SwiftOnFileFS` is already created.
1. Create a container named `unified_access` with unified file and object access storage policy using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $auth_token"
-X PUT http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df/unified_access/
-H "X-Storage-Policy: SwiftOnFileFS"
```

In this command, `http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df/` is the endpoint URL for a project (tenant) using which a container `unified_access` is created with `SwiftOnFileFS` as the storage policy.

2. Upload an object in the container associated with the unified file and object access storage policy using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $auth_token"
-X PUT http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/object1
--data-binary @imageA.jpg
```

3. Download object residing in the unified file and object access container using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $auth_token"
-X GET http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/samplefile.txt
```

4. List the contents of the unified file and object access container using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $auth_token"
-X GET http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/
```

Configuration files for IBM Spectrum Scale for object storage

Use the following information to manage options in configuration files that are used for IBM Spectrum Scale for object storage including the unified file and object access feature. These configuration files are located in the `/etc/swift` directory.

For information on changing an option in a configuration file, see “Changing options in configuration files” on page 155.

object-server-sof.conf

- Contains identity management modes for unified file and object access (**id_mgmt**)
- Contains AD domain name (**ad_domain**) if AD is configured

Table 11. Configurable options for [DEFAULT] in `object-server-sof.conf`

Configuration option = Default value	Description
id_mgmt = local_mode	Defines the object server behavior while assigning user or group ownership to newly created objects, when those are accessed using the file interface. The allowed values are <code>local_mode</code> and <code>unified_mode</code> . With <code>local_mode</code> , the new objects are owned by the <code>swift</code> user. In <code>unified_mode</code> , the identity of the user making the PUT request is fetched from the configured directory server.
ad_domain	When using Active Directory (AD), defines the AD domain from which the user identity should be fetched when object server is operating in the <code>unified_mode</code> identity management mode. Note: When you clean up object authentication, you must manually remove this entry. For more information, see “Configuring authentication and setting identity management modes for unified file and object access” on page 143.
tempfile_prefix = .ibmtmp_	The prefix to be used for the temporary file being created when a file being uploaded.
disable_fallocate = true	Overrides the default swift fallocate behavior, and relies on the GPFS fallocate features, excludes 'fast fail' checks.
disk_chunk_size = 65536	The size of chunks to read/write to disk (needs be equal to the file system block size).

Table 11. Configurable options for [DEFAULT] in object-server-sof.conf (continued)

Configuration option = Default value	Description
network_chunk_size = 65536	The size of chunks to read/write over the network (needs be equal to the file system block size).
log_statsd_host = localhost	If not set, the StatsD feature is disabled.
log_statsd_port = 8125	The port number for the StatsD server.
log_statsd_default_sample_rate = 1.0	Defines the probability of sending a sample for any given event or timing measurement.
log_statsd_sample_rate_factor = 1.0	Not recommended to set this to a value less than 1.0. If the frequency of logging is too high, tune the log_statsd_default_sample_rate instead.
log_statsd_metric_prefix =	The prefix that is added to every metric sent to the StatsD server.
retain_acl = yes	Specifies whether or not to copy the ACL from an existing object. Allowed values are yes or no.
retain_winattr = yes	Specifies whether or not to copy the Windows attributes from an existing object. Allowed values are yes or no.
retain_xattr = yes	Specifies whether or not to copy the extended attributes for the user namespace from an existing object. Allowed values are yes or no.
retain_owner = yes	Specifies whether or not to copy the UID/GID owners from an existing object. Allowed values are yes or no.

Note: Files with the .ibtmp prefix or the one configured in the object-server-sof.conf configuration file are not objectized.

When you set the retain_* options to yes, the following attributes are retained:

- The extended attributes in the user namespace except for the **user.swift.metadata** key which contains swift metadata and it is expected to be new.
- Windows attributes

When you set the retain_* options to yes, the following attributes are not retained:

- Extended attributes in system, security, and trusted namespaces.

Note: These attributes are not retained in an object's copy object operation also.

Retaining ACLs, Windows attributes, file extended attributes, and ownership, when an object is PUT over an existing object in unified file and object access enabled containers depends on your specific use case and your discretion. For example, if you are using object and file access to refer to the same data content in such a way that the object protocol might completely replace the data content in such that it might be completely new content from the file interface as well, then you might choose to not retain the existing file ACL and extended attributes. For such a use case, you might change the default values to not to retain the file ACLs, extended attributes, and ownership.

Note: If you are unsure about whether to retain these attributes or not, you might want to use the default values of retaining ACLs, Windows attributes, file extended attributes, and ownership. The default values in this case are more aligned with the expected behavior in a multiprotocol setup.

spectrum-scale-object.conf

- Contains cluster or fileset configuration information
- Unique to a site

Table 12. Configurable options for [capabilities] in spectrum-scale-object.conf

Configuration option = Default value	Description
file-access-enabled = false	The state for the file-access capability. It can be either true or false.
multi-region-enabled = true	The state for the multi-region capability. This option cannot be changed.
s3-enabled = true	The state for the s3 capability. This option cannot be changed.

spectrum-scale-objectizer.conf

.

Contains the **ibmobjectizer** service configuration information

Table 13. Configuration options for [DEFAULT] in spectrum-scale-objectizer.conf.

Configuration option = Default value	Description
objectization_tmp_dir =	The temporary directory to be used by ibmobjectizer . This must be a path on any GPFS file system. The default value is autofilled with the path of the base file system for object.
objectization_threads = 24	The maximum number of threads that ibmobjectizer will spawn on a node.
batch_size = 100	The maximum number of files that ibmobjectizer will process in a thread.
objectization_interval = 1800	The time interval, in seconds, between the completion of an objectization cycle and the beginning of the next objectization cycle.

Table 14. Configuration options for [IBMOBJECTIZER-LOGGER] in spectrum-scale-objectizer.conf.

Configuration option = Default value	Description
log_level = INFO	The logging level. Allowed value is one of the following: INFO, DEBUG, WARN, ERROR

Changing options in configuration files

You can use the **mmobj config change** command to change the values of the options in the configuration files. For example:

- Change the value of an option in the [DEFAULT] section of the object-server-sof.conf file as follows:

```
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property OPTIONNAME --value NEWVALUE
```
- Change the value of an option in the [IBMOBJECTIZER-LOGGER] section of the spectrum-scale-objectizer.conf file as follows:

```
mmobj config change --ccrfile spectrum-scale-objectizer.conf --section IBMOBJECTIZER-LOGGER
--property OPTIONNAME --value NEWVALUE
```

Note: Only some options are configurable. If an option cannot be changed, it is mentioned in the respective option description.

Attention: When a configuration file is changed using these commands, it takes several seconds for the changes to be synced across the whole cluster depending on the size of the cluster. Therefore, when executing multiple commands to change configuration files, you must plan for an adequate time interval between the execution of these commands.

Chapter 7. Managing disks

Use the following information to manage disks in IBM Spectrum Scale

Disks can have connectivity to each node in the cluster, be managed by network shared disk servers, or a combination of the two. For more information, see “mmcrnsd command” on page 426. Also see, *Network Shared Disk (NSD) creation considerations* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Note: A LUN provided by a storage subsystem is a disk for the purposes of this documentation, even if the LUN is made up of multiple physical disks.

The disk related tasks performed on a GPFS file system include:

1. “Displaying disks in a GPFS cluster”
2. “Adding disks to a file system” on page 158
3. “Deleting disks from a file system” on page 158
4. “Replacing disks in a GPFS file system” on page 160
5. “Additional considerations for managing disks” on page 161
6. “Displaying GPFS disk states” on page 162
7. “Changing GPFS disk states and parameters” on page 163
8. “Changing your NSD configuration” on page 165
9. “Changing NSD server usage and failback” on page 166
10. “Enabling and disabling Persistent Reserve” on page 166

Displaying disks in a GPFS cluster

You can display the disks that belong to your GPFS cluster by issuing the **mmfnsd** command.

The default is to display information for all disks defined to the cluster (**-a**). Otherwise, you may choose to display the information for a particular file system (**-f**) or for all disks which do not belong to any file system (**-F**).

To display the default information for all of the NSDs belonging to the cluster, enter:

```
mmfnsd
```

The system displays information similar to:

```
File system  Disk name  NSD servers
-----
fs2          hd3n97    c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd4n97    c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd5n98    c5n98g.ppd.pok.ibm.com,c5n97g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd6n98    c5n98g.ppd.pok.ibm.com,c5n97g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          sdbnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdcnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sddnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdensd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdgnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdfnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdhnsd    c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
(fs free disk) hd2n97    c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com
```

To find out the local device names for the disks, use the **mmlsnsd** command with the **-m** option. For example, issuing **mmlsnsd -m** produces output similar to this:

Disk name	NSD volume ID	Device	Node name	Remarks
hd2n97	0972846145C8E924	/dev/hdisk2	c5n97g.ppd.pok.ibm.com	server node
hd2n97	0972846145C8E924	/dev/hdisk2	c5n98g.ppd.pok.ibm.com	server node
hd3n97	0972846145C8E927	/dev/hdisk3	c5n97g.ppd.pok.ibm.com	server node
hd3n97	0972846145C8E927	/dev/hdisk3	c5n98g.ppd.pok.ibm.com	server node
hd4n97	0972846145C8E92A	/dev/hdisk4	c5n97g.ppd.pok.ibm.com	server node
hd4n97	0972846145C8E92A	/dev/hdisk4	c5n98g.ppd.pok.ibm.com	server node
hd5n98	0972846245EB501C	/dev/hdisk5	c5n97g.ppd.pok.ibm.com	server node
hd5n98	0972846245EB501C	/dev/hdisk5	c5n98g.ppd.pok.ibm.com	server node
hd6n98	0972846245DB3AD8	/dev/hdisk6	c5n97g.ppd.pok.ibm.com	server node
hd6n98	0972846245DB3AD8	/dev/hdisk6	c5n98g.ppd.pok.ibm.com	server node

Adding disks to a file system

Many file systems grow rapidly, so after creating a file system you might decide that more disk space is required.

Storage in a file system is divided in storage pools. The maximum size of any one disk that can be added to an existing storage pool is set approximately to the sum of the disk sizes when the storage pool is created. The actual value is shown in the **mmdf** command output.

Once a storage pool is created, the maximum size *cannot* be altered. However, you can create a new pool with larger disks, and then move data from the old pool to the new one.

When establishing a storage pool and when adding disks later to an existing storage pool, you should try to keep the sizes of the disks fairly uniform. GPFS allocates blocks round robin, and as the utilization level rises on all disks, the small ones will fill up first and all files created after that will be spread across fewer disks, which reduces the amount of prefetch that can be done for those files.

To add disks to a GPFS file system, first decide if you will:

1. Create new disks using the **mmcrnsd** command.
In this case, you must also decide whether to create a new set of NSD and pools stanzas or use the rewritten NSD and pool stanzas that the **mmcrnsd** command produces. In a rewritten file, the disk usage, failure group, and storage pool values are the same as the values that are specified in the **mmcrnsd** command.
2. Select disks no longer in use in any file system. Issue the **mmlsnsd -F** command to display the available disks.

The disk may then be added to the file system using the stanza file as input to the **mmadddisk** command.

Note: Rebalancing of files is an I/O intensive and time consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance your file system over time, without the cost of the rebalancing.

For more information, see “mmadddisk command” on page 239 and “mmcrnsd command” on page 426.

Deleting disks from a file system

Before deleting a disk use the **mmdf** command to determine whether there is enough free space on the remaining disks to store the file system.

Note: See “Querying file system space” on page 43 for more information about diagnosing space problems.

Consider how fragmentation may increase your storage requirements, especially when the file system contains a large number of small files. A margin of 150 percent of the size of the disks being deleted should be sufficient to allow for fragmentation when small files predominate. For example, in order to delete a 400 GB disk from your file system, which contains user home directories with small files, you should first determine that the other disks in the file system contain a total of 600 GB of free space.

If you do not replicate your file system data, you should rebalance the file system using the **mmrestripefs -b** command. If you replicate your file system data, run the **mmrestripefs -r** command after the disk has been deleted. This ensures that all data will still exist with correct replication after the disk is deleted. The **mmdeldisk** command only migrates data that would otherwise be lost, not data that will be left in a single copy.

Note: Rebalancing of files is an I/O intensive and time consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance your file system over time, without the cost of the rebalancing.

Do not delete stopped disks, if at all possible. Start any stopped disk before attempting to delete it from the file system. If the disk cannot be started you will have to consider it permanently damaged. You will need to delete the disk using the appropriate **mmdeldisk** options. If metadata was stored on the disk, you will need to execute **mmfsck** in offline mode afterwards.. For more information on handling this situation, see *NSD and underlying disk subsystem failures* in the *IBM Spectrum Scale: Problem Determination Guide*.

When deleting disks from a file system, the disks may or may not be available. If the disks being deleted are still available, GPFS moves all of the data from those disks to the disks remaining in the file system. However, if the disks being deleted are damaged, either partially or permanently, it is not possible to move all of the data and you will receive I/O errors during the deletion process. For instructions on how to handle damaged disks, see *Disk media failure* in the *IBM Spectrum Scale: Problem Determination Guide*.

Specify the file system and the names of one or more disks to delete with the **mmdeldisk** command. For example, to delete the disk **hd2n97** from the file system **fs2** enter:

```
mmdeldisk fs2 hd2n97
```

The system displays information similar to:

```
Deleting disks ...
Scanning system storage pool
Scanning file system metadata, phase 1 ...
19 % complete on Fri Mar 16 23:23:50 2012
100 % complete on Fri Mar 16 23:23:51 2012
Scan completed successfully.
Scanning file system metadata, phase 2 ...
46 % complete on Fri Mar 16 23:23:55 2012
93 % complete on Fri Mar 16 23:23:58 2012
100 % complete on Fri Mar 16 23:23:58 2012
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
19.50 % complete on Fri Mar 16 23:24:25 2012 ( 35777 inodes 1207 MB)
47.92 % complete on Fri Mar 16 23:24:49 2012 ( 199955 inodes 2966 MB)
50.05 % complete on Fri Mar 16 23:25:09 2012 ( 235356 inodes 3098 MB)
53.09 % complete on Fri Mar 16 23:25:31 2012 ( 261831 inodes 3286 MB)
55.12 % complete on Fri Mar 16 23:25:51 2012 ( 283815 inodes 3412 MB)
63.25 % complete on Fri Mar 16 23:26:12 2012 ( 319236 inodes 3915 MB)
```

```

63.27 % complete on Fri Mar 16 23:26:33 2012 ( 382031 inodes 6223 MB)
63.29 % complete on Fri Mar 16 23:27:03 2012 ( 699858 inodes 9739 MB)
100.00 % complete on Fri Mar 16 23:27:35 2012
Scan completed successfully.
Checking Allocation Map for storage pool 'system'
17 % complete on Fri Mar 16 23:27:42 2012
31 % complete on Fri Mar 16 23:27:47 2012
48 % complete on Fri Mar 16 23:27:52 2012
62 % complete on Fri Mar 16 23:27:57 2012
76 % complete on Fri Mar 16 23:28:02 2012
90 % complete on Fri Mar 16 23:28:07 2012
100 % complete on Fri Mar 16 23:28:08 2012
tsdelldisk completed.
mmdeldisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```

For syntax and usage information, refer to “mmdeldisk command” on page 449.

Replacing disks in a GPFS file system

Replacing an existing disk in a GPFS file system with a new one is the same as performing a delete disk operation followed by an add disk. However, this operation eliminates the need to restripe the file system following the separate delete disk and add disk operations as data is automatically moved to the new disk.

When replacing disks in a GPFS file system, first decide if you will:

1. Create new disks using the **mmcrnsd** command.
In this case, you must also decide whether to create a new set of NSD and pools stanzas or use the rewritten NSD and pool stanzas that the **mmcrnsd** command produces. In a rewritten file, the disk usage, failure group, and storage pool values are the same as the values that are specified in the **mmcrnsd** command.
2. Select NSDs no longer in use by another GPFS file system. Issue the **mmlnsd -F** command to display the available disks.

To replace a disk in the file system, use the **mmrpldisk** command. For example, to replace the NSD **hd3n97** in file system **fs2** with the existing NSD **hd2n97**, which is no longer in use by another file system, enter:

```
mmrpldisk fs2 hd3n97 hd2n97
```

The system displays information similar to:

```
Replacing hd3n97 ...
```

```

The following disks of fs2 will be formatted on node c33f2in01:
hd2n97: size 571398144 KB
Extending Allocation Map
Checking Allocation Map for storage pool 'system'
9 % complete on Fri Mar 16 23:33:29 2012
23 % complete on Fri Mar 16 23:33:34 2012
37 % complete on Fri Mar 16 23:33:40 2012
52 % complete on Fri Mar 16 23:33:45 2012
66 % complete on Fri Mar 16 23:33:50 2012
83 % complete on Fri Mar 16 23:33:55 2012
98 % complete on Fri Mar 16 23:34:00 2012
100 % complete on Fri Mar 16 23:34:00 2012
Completed adding disks to file system fs2.
Scanning system storage pool
Scanning file system metadata, phase 1 ...
13 % complete on Fri Mar 16 23:34:19 2012
100 % complete on Fri Mar 16 23:34:22 2012
Scan completed successfully.
Scanning file system metadata, phase 2 ...

```

```

29 % complete on Fri Mar 16 23:34:26 2012
67 % complete on Fri Mar 16 23:34:29 2012
100 % complete on Fri Mar 16 23:34:32 2012
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
8.21 % complete on Fri Mar 16 23:34:54 2012 ( 37741 inodes 770 MB)
22.65 % complete on Fri Mar 16 23:35:14 2012 ( 40182 inodes 2124 MB)
32.95 % complete on Fri Mar 16 23:35:34 2012 ( 160837 inodes 3090 MB)
35.15 % complete on Fri Mar 16 23:35:57 2012 ( 227991 inodes 3296 MB)
36.34 % complete on Fri Mar 16 23:36:17 2012 ( 265748 inodes 3408 MB)
37.34 % complete on Fri Mar 16 23:36:38 2012 ( 284398 inodes 3502 MB)
46.07 % complete on Fri Mar 16 23:37:04 2012 ( 310636 inodes 4320 MB)
61.41 % complete on Fri Mar 16 23:37:25 2012 ( 315141 inodes 5759 MB)
87.04 % complete on Fri Mar 16 23:37:50 2012 ( 350241 inodes 8163 MB)
87.06 % complete on Fri Mar 16 23:38:11 2012 ( 370562 inodes 10136 MB)
87.08 % complete on Fri Mar 16 23:38:42 2012 ( 392561 inodes 11982 MB)
87.10 % complete on Fri Mar 16 23:39:22 2012 ( 401049 inodes 13195 MB)
87.12 % complete on Fri Mar 16 23:40:14 2012 ( 1100590 inodes 15685 MB)
100.00 % complete on Fri Mar 16 23:40:57 2012
Scan completed successfully.
Checking Allocation Map for storage pool 'system'
10 % complete on Fri Mar 16 23:41:02 2012
26 % complete on Fri Mar 16 23:41:07 2012
33 % complete on Fri Mar 16 23:41:12 2012
44 % complete on Fri Mar 16 23:41:17 2012
68 % complete on Fri Mar 16 23:41:22 2012
100 % complete on Fri Mar 16 23:41:25 2012
Done
mmrpldisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```

Note: If you attempt to replace a stopped disk and the file system is not replicated, the attempt will fail.

However, you can replace a stopped disk if the file system is replicated. You can do so in one of the following ways:

- Deletion, addition, and rebalancing method:
 1. Use the **mmdeledisk** command to delete the stopped disk from the file system.
 2. Use the **mmaddisk** command to add a replacement disk.
 3. Use the **mmrestripefs -b** command to rebalance the file system.

While this method requires rebalancing, it returns the system to a protected state faster (because it can use all of the remaining disks to create new replicas), thereby reducing the possibility of losing data.

—Or—

- Direct replacement method:

Use the **mmrpldisk** command to directly replace the stopped disk.

The **mmrpldisk** command only runs at single disk speed because all data being moved must be written to the replacement disk. The data is vulnerable while the command is running, and should a second failure occur before the command completes, it is likely that some data will be lost.

For more information on handling this situation, see *Disk media failure* in the *IBM Spectrum Scale: Problem Determination Guide*.

Additional considerations for managing disks

If you delete, replace, or suspend a disk with strict replication enforced, you may receive an ENOSPC error when you create or append data to an existing file.

If you need to delete, replace, or suspend a disk and you need to write new data while the disk is offline, you can disable strict replication before you perform the disk action. However, data written while replication is disabled will not be properly replicated. Therefore, after you perform the disk action, you must re-enable strict replication and run the **mmrestripefs -r** command. To determine if a file system has strict replication enforced, issue the **mmlsfs -K** command.

Displaying GPFS disk states

You can display the current state of one or more disks in your file system by issuing the **mmlsdisk** command.

The information includes parameters that were specified on the **mmcrfs** command, and the current availability and status of the disks. For example, to display the current status of the disk **hd8vsdn100** in the file system **fs1**, enter:

```
mmlsdisk fs1 -d hd8vsdn100
```

Status is displayed in a format similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
hd8vsdn100	nsd	512	1	no	yes	ready	up	spl

For syntax and usage information, see “mmlsdisk command” on page 528.

Disk availability

The following information lists the possible values of disk availability, and what they mean.

A disk's availability determines whether GPFS is able to read and write to the disk. There are four possible values for availability:

up The disk is available to GPFS for normal **read** and **write** operations.

down No **read** and **write** operations can be performed on the disk.

recovering

An intermediate state for disks coming up, during which GPFS verifies and corrects data. **write** operations can be performed while a disk is in this state, but **read** operations cannot (because data on the disk being recovered might be stale until the **mmchdisk start** command completes).

unrecovered

The disk was not successfully brought up.

Disk availability is automatically changed from **up** to **down** when GPFS detects repeated I/O errors. You can also change the availability of a disk by issuing the **mmchdisk** command.

Disk status

The following information lists the possible values for disk status, and what they mean.

Disk status controls data placement and migration. Status changes as a result of a pending delete operation, or when the **mmchdisk** command is issued to allow file rebalancing or re-replicating prior to disk replacement or deletion.

Disk status has seven possible values, but four are transitional:

ready Normal status.

suspended

or

to be emptied

Indicates that data is to be migrated off this disk.

being emptied

Transitional status in effect while a disk deletion is pending.

emptied

Indicates that data is already migrated off this disk.

replacing

Transitional status in effect for old disk while replacement is pending.

replacement

Transitional status in effect for new disk while replacement is pending.

GPFS allocates space only on disks with a status of **ready** or **replacement**.

GPFS migrates data off disks with a status of **being emptied**, **replacing**, **to be emptied**, or **suspended** onto disks with a status of **ready** or **replacement**. During disk deletion or replacement, data is automatically migrated as part of the operation. Issue the **mmrestripefs** command to initiate data migration from a suspended disk.

See “Deleting disks from a file system” on page 158, “Replacing disks in a GPFS file system” on page 160, and “Restripping a GPFS file system” on page 42.

Changing GPFS disk states and parameters

You might find it necessary to change a disk's state if there is some indication of disk failure or if you need to restripe the file system.

Refer to “Displaying GPFS disk states” on page 162 for a detailed description of disk states. You can change both the availability and status of a disk using the **mmchdisk** command:

- Change disk availability using the **mmchdisk** command and the **stop** and **start** options
- Change disk status using the **mmchdisk** command and the **suspend** and **resume** options.

Issue the **mmchdisk** command with one of the following four options to change disk state:

resume

Informs GPFS that a disk previously suspended is now available for allocating new space. Resume a disk only when you've suspended it and decided not to delete or replace it. If the disk is currently in a stopped state, it remains stopped until you specify the **start** option. Otherwise, normal **read** and **write** access to the disk resumes.

start

Informs GPFS that a disk previously stopped is now accessible. GPFS does this by first changing the disk availability from **down** to **recovering**. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was down) are repaired. If this operation is successful, the availability is then changed to **up**.

If the metadata scan fails, availability is set to **unrecovered**. This could occur if other disks remain in **recovering** or an I/O error has occurred. Repair all disks and paths to disks. It is recommended that you run **mmfsck** at this point (For more information, see “mmfsck command” on page 487.). The metadata scan can then be re-initiated at a later time by issuing the **mmchdisk start** command again.

If more than one disk in the file system is down, they should all be started at the same time by using the **-a** option. If you start them separately and metadata is stored on any disk that remains down, the **mmchdisk start** command fails.

stop

Instructs GPFS to stop any attempts to access the specified disk. Use this option to inform GPFS that

a disk has failed or is currently inaccessible because of maintenance. A disk's availability remains **down** until it is explicitly started with the **start** option.

suspend
or
empty

Instructs GPFS to stop allocating space on the specified disk. Place a disk in this state prior to disk deletion or replacement. This is a user-initiated state that GPFS will never use without an explicit command to change disk state.

Note: A disk remains suspended until it is explicitly resumed. Restarting GPFS or rebooting nodes does not restore normal access to a suspended disk.

The empty option is similar to the suspend option. For releases prior to GPFS 4.1.1, the output of the **mmfsdisk** command displays the status as suspended, as shown in the following example.

For example, to suspend the **hd8vsdn100** disk in the file system **fs1**, enter:

```
mmchdisk fs1 suspend -d hd8vsdn100
```

To confirm the change, enter:

```
mmfsdisk fs1 -d hd8vsdn100
```

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
hd8vsdn100	nsd	512	7	yes	yes	suspended	up	system

From GPFS 4.1.1 onwards, the status in the **mmfsdisk** command is displayed as to be emptied, as shown in the following example:

For example, to empty the **gpfs1nsd** disk in the file system **fs1**, enter:

```
mmchdisk fs1 empty -d gpfs1nsd
```

To confirm the change, enter:

```
mmfsdisk fs1 -d gpfs1nsd
```

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	disk id	storage pool	remarks
gpfs1nsd	nsd	512	-1	Yes	Yes	to be emptied	up	1	system	
gpfs2nsd	nsd	512	-1	Yes	Yes	to be emptied	up	2	system	desc

You can also use the **mmchdisk** command with the **change** option to change the *Disk Usage* and *Failure Group* parameters for one or more disks in a GPFS file system. This can be useful in situations where, for example, a file system that contains only RAID disks is being upgraded to add conventional disks that are better suited to storing metadata. After adding the disks using the **mmaddisk** command, the metadata currently stored on the RAID disks would have to be moved to the new disks to achieve the desired performance improvement. To accomplish this, first the **mmchdisk change** command would be issued to change the *Disk Usage* parameter for the RAID disks to **dataOnly**. Then the **mmrestripefs** command would be used to restripe the metadata off the RAID device and onto the conventional disks.

For example, to specify that metadata should no longer be stored on disk **hd8vsdn100**, enter:

```
mmchdisk fs1 change -d "hd8vsdn100:::dataOnly"
```

To confirm the change, enter:

```
mmfsdisk fs1 -d hd8vsdn100
```

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
hd8vsdn100	nsd	512	1	no	yes	ready	up	sp1

For complete usage information, see the “mmchdisk command” on page 354 and the “mmlsdisk command” on page 528.

Changing your NSD configuration

Use the following steps to change the NSD configuration.

Once your NSDs have been created, you may change the configuration attributes as your system requirements change. For more information about creating NSDs, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the “mmcrnsd command” on page 426.

By issuing the **mmchnsd** command you can:

- Specify up to eight servers for an NSD that does not have one.
- Change the NSD server nodes specified in the server list.
- Delete the server list. The disk must now be SAN-attached to all nodes in the cluster on which the file system will be mounted.

You must follow these rules when changing NSDs:

- Identify the disks by the NSD names that were given to them by the **mmcrnsd** command.
- Explicitly specify values for all NSD servers in the list, even if you are only changing one of the values.
- Unmount the file system that contains the NSD being changed prior to issuing the **mmchnsd** command.
- Connect the NSD to the new nodes prior to issuing the **mmchnsd** command.
- **mmchnsd** cannot change the *DiskUsage* or *FailureGroup* for an NSD. Use the **mmchdisk** command to change these attributes.
- To move a disk from one storage pool to another, use the **mmdeldisk** and **mmadddisk** commands.
- You cannot change the name of the NSD.

For example, to assign node **k145n07** as an NSD server for disk **gpfs47nsd**:

1. Make sure that **k145n07** is not already assigned to the server list by issuing the **mmlsnsd** command.

```
mmlsnsd -d "gpfs47nsd"
```

The system displays information similar to:

File system	Disk name	NSD server nodes
fs1	gpfs47nsd	k145n09

2. Unmount the file system on all nodes and ensure that the disk is connected to the new node (**k145n07**).
3. Issue the **mmchnsd** command:

```
mmchnsd "gpfs47nsd:k145n09,k145n07"
```
4. Verify the changes by issuing the **mmlsnsd** command:

```
mmlsnsd -d gpfs47nsd
```

The system displays information similar to:

File system	Disk name	NSD servers
fs2	gpfs47nsd	k145n09.ppd.pok.ibm.com,k145n07.ppd.pok.ibm.com

Changing NSD server usage and failback

GPFS determines if a node has physical or virtual connectivity to an underlying NSD disk through a sequence of commands invoked from the GPFS daemon. This determination is called disk discovery and occurs at both initial GPFS startup as well as whenever a file system is mounted.

The default order of access used in disk discovery:

1. Local block device interfaces for SAN, SCSI or IDE disks
2. NSD servers

The **useNSDserver** file system mount option can be used to set the order of access used in disk discovery, and limit or eliminate switching from local access to NSD server access, or the other way around. This option is specified using the **-o** flag of the **mmmount**, **mount**, **mmchfs**, and **mmremotefs** commands, and has one of these values:

always

Always access the disk using the NSD server.

asfound

Access the disk as found (the first time the disk was accessed). No change of disk access from local to NSD server, or the other way around, is performed by GPFS.

asneeded

Access the disk any way possible. This is the default.

never Always use local disk access.

For example, to always use the NSD server when mounting file system **fs1**, issue this command:

```
mmmount fs1 -o useNSDserver=always
```

To change the disk discovery of a file system that is already mounted: cleanly unmount it, wait for the unmount to complete, and then mount the file system using the desired **-o useNSDserver** option.

Enabling and disabling Persistent Reserve

GPFS can use Persistent Reserve (PR) functionality to improve failover times (with some restrictions).

The following restrictions apply to the use of PR:

- PR is supported on both AIX and Linux nodes. However, note the following:
 - If the disks have defined NSD servers, then the NSD server nodes must all be running AIX, or they must all be running Linux.
 - If the disks are SAN-attached to all nodes, then the SAN-attached nodes in the cluster must all be running AIX, or they must all be running Linux.
- The disk subsystems must support PR
- GPFS supports a mix of PR disks and other disks. However, you will only realize improved failover times if **all** disks in the cluster support PR.
- GPFS only supports PR in the local cluster. Remote mounts must access the disks through an NSD server.
- When you enable or disable PR, you must stop GPFS on all nodes.
- Before enabling PR, make sure all disks are in the same initial state.

To enable (or disable) Persistent Reserve, issue the command:

```
mmchconfig usePersistentReserve={yes|no}
```

For fast recovery times with Persistent Reserve, you should also set the *failureDetectionTime* configuration parameter. For fast recovery, a recommended value would be 10. You can set this by issuing the command:

```
mmchconfig failureDetectionTime=10
```

To determine if the disks on the servers and the disks of a specific node have PR enabled, issue the following command from the node:

```
mm1snsd -X
```

The system responds with something similar to:

Disk name	NSD volume ID	Device	Devtype	Node name	Remarks
gpfs10nsd	09725E5E43035A99	/dev/hdisk6	hdisk	k155n14.kgn.ibm.com	server node,pr=yes
gpfs10nsd	09725E5E43035A99	/dev/hdisk8	hdisk	k155n16.kgn.ibm.com	server node,pr=yes
gpfs10nsd	09725E5E43035A99	/dev/hdisk6	hdisk	k155n17.kgn.ibm.com	directly attached pr=yes

If the GPFS daemon has been started on all the nodes in the cluster and the file system has been mounted on all nodes that have direct access to the disks, then **pr=yes** should be on all hdisks. If you do not see this, there is a problem. Refer to the *IBM Spectrum Scale: Problem Determination Guide* for additional information on Persistent Reserve errors.

Chapter 8. Managing GPFS quotas

The GPFS quota system helps you to control the allocation of files and data blocks in a file system.

GPFS quotas can be defined for:

- Individual users
- Groups of users
- Individual filesets

Quotas are enabled by the system administrator when control over the amount of space used by the individual users, groups of users, or individual filesets is required. By default, user and group quota limits are enforced across the entire file system. Optionally, the scope of quota enforcement can be limited to an individual fileset boundaries.

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Files > Quotas**.

Quota related tasks include:

1. “Enabling and disabling GPFS quota management”
2. “Default quotas” on page 170
3. “Explicitly establishing and changing quotas” on page 173
4. “Checking quotas” on page 176
5. “Listing quotas” on page 177
6. “Activating quota limit checking” on page 178
7. “Deactivating quota limit checking” on page 178
8. “Changing the scope of quota limit checking” on page 179
9. “Creating file system quota reports” on page 179
10. “Restoring quota files” on page 180

For GPFS fileset quotas, see *Filesets* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Note: Windows nodes may be included in clusters that use GPFS quotas; however, Windows nodes do not support the quota commands.

Enabling and disabling GPFS quota management

You can enable GPFS quota management on new or existing GPFS file systems, establish quota values, and disable quota management by following the steps in this topic.

To enable GPFS quota management on a new GPFS file system:

1. Specify the **-Q yes** option on the **mmcrfs** command. This option automatically activates quota enforcement whenever the file system is mounted. If you want the scope of quota limit enforcement to be based on individual filesets (rather than the entire file system), also specify the **--perfileset-quota** option on the **mmcrfs** command.
2. Mount the file system.
3. Issue the **mmedquota** or **mmsetquota** command to explicitly set quota values for users, groups, or filesets. See “Explicitly establishing and changing quotas” on page 173.

To enable GPFS quota management on an existing GPFS file system:

1. Run the **mmchfs -Q yes** command. This command automatically activates quota enforcement whenever the file system is mounted or activates all subsequent mounts following the new quota setting if the file system is not mounted. If you want the scope of quota limit enforcement to be based on individual filesets (rather than the entire file system), also specify the **--perfileset-quota** option on the **mmcrfs** command.

If an online **mmchfs -Q yes/no** command fails or is interrupted for any reason, **mmcheckquota** or **mmchfs -Q yes/no** must be rerun so that quota configuration for all nodes in the cluster will be brought into a consistent state.

All subsequent mounts will follow the new quota setting.

Note: The **perfileset-quota** cannot be enabled online in GPFS 4.1.

2. Compile inode and disk block statistics using the **mmcheckquota** command. See “Checking quotas” on page 176. The values obtained can be used to establish realistic quota values when issuing the **mmedquota** or **mmsetquota** command.
3. Issue the **mmedquota** or **mmsetquota** command to explicitly set quota values for users, groups, or filesets. See “Explicitly establishing and changing quotas” on page 173.

Once GPFS quota management has been enabled, you may establish quota values by:

- Setting default quotas for all new users, groups of users, or filesets.
- Explicitly establishing or changing quotas for users, groups of users, or filesets.
- Using the **gpfs_quotactl()** subroutine.

To disable quota management, run the **mmchfs -Q no** command. All subsequent mounts will obey the new quota setting.

For complete usage information, see the “**mmcheckquota** command” on page 361, the “**mmchfs** command” on page 371, the “**mmcrfs** command” on page 414, and the “**mmedquota** command” on page 481. For additional information on quotas, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Default quotas

Default quota limits can be set for new users, groups, and filesets for a specified file system. Default quota limits can also be applied at a more granular level for new users and groups in a specified fileset.

When default quotas are managed at the fileset level, those quotas have a higher priority than those set at the file system level. If the status of the fileset-level defaults for one fileset is **Initial**, they will inherit default limits from global fileset-level defaults. The status of newly added fileset-level default quotas can be one of the following:

Initial When the fileset is created, it will be in this state. All user and group quota accounts under the fileset will not follow the fileset defaults.

Quota on

All user and group quota accounts under the fileset that are created later will follow the fileset quota limits.

Quota off

All user and group quota accounts under the fileset that are created later will not follow the fileset quota limits. The users and groups will follow global fileset-level defaults if they are valid. If those defaults are not valid, the status will be initial.

Specific default quota recommendations for protocols:

- Since the protocols may have vastly different fileset requirements, it is not recommended to use default quotas at the fileset level. Rather, set explicit quotas and limits for each fileset in use by any and all protocols on a case-by-case basis.
- NFS: Prepare a default quota stanza file template, and at NFS export creation time, apply the default user or group quotas to the export path (assuming the export is an independent fileset) using per-fileset default quotas.
- SMB: Prepare a default quota stanza file template, and at SMB share creation time, apply the default user or group quotas to the share path (assuming the export is an independent fileset) using per-fileset default quotas.
- Object: IBM recommends using a single independent fileset, `objectfs`, for the object container. See *IBM Redpaper: A Deployment Guide for IBM Spectrum Scale Object* for details. With regard to quotas, here are the relevant sections from the Redpaper™:
 - GPFs quotas: The amount of disk space and the number of inodes that are assigned as upper limits for a specified user, group of users, or fileset. With OpenStack Swift, GPFs user quotas are not used; instead, the system relies on OpenStack Swift quotas to provide a similar type of service. However, GPFs fileset quotas can still be defined (for example, for inodes, to limit the resources that are consumed by the fileset). See *Chapter 1.3 Key concepts and terminology* of the IBM Redpaper for details.
 - Swift quotas: Allows specification of the amount of disk space or number of objects that can be consumed by either an account (and subsequently all of its containers) or to an individual container. The interaction between Swift quotas and GPFs quotas are described in more detail in *Chapter 6 Swift feature overview* and *Chapter 1.3 Key concepts and terminology* of the IBM Redpaper.
 - Quotas: Swift quotas allow a specific amount of disk capacity to be allocated to either containers or accounts by using Swift quotas. They also allow a limit on the maximum number of objects to be specified for containers or accounts. See *Chapter 6 Swift feature overview* of the IBM Redpaper for details.

Note: Although GPFs quotas do not explicitly interact with Swift quotas, it still might be useful to employ GPFs quotas to limit the amount of space or the number of inodes that is consumed by the object store. To do this, define GPFs quotas on the top-level independent fileset by specifying the maximum size or maximum inode usage that the object store can consume. See *Chapter 6 Swift feature overview* of the IBM Redpaper for details.

To enable default quota values:

1. Ensure the file system is configured correctly to use quotas:
 - a. The **-Q yes** option must be in effect for the file system.
 - b. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect.

Note: If **--perfileset-quota** is in effect, all users and groups in the fileset **root** will not be impacted by default quota unless they are explicitly set.

The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmfsfs** command to display the current settings of these quota options.

2. Enable default quotas with the **mmdefquotaon** command.
3. Specify default quota values for new users, groups, and filesets by issuing the **mmdefedquota** command using a default quota stanza file. A single invocation of the **mmsetquota** command using a quota stanza file can perform the following operations:
 - Set default user quotas on a file system.
 - Set default group quotas on a file system.
 - Set a default perfileset user quota on a fileset (if **--perfileset-quota** is in effect).

The stanza file `/tmp/defaultQuotaExample` may look like this:

```
%quota:  
device=fs1  
command=setDefaultQuota  
type=USR  
blockQuota=25G  
blockLimit=30G  
filesQuota=10K  
filesLimit=11K
```

```
%quota:  
device=fs1  
command=setDefaultQuota  
type=GRP  
blockQuota=75G  
blockLimit=90G  
filesQuota=30K  
filesLimit=33K
```

```
%quota:  
device=fs1  
command=setDefaultQuota  
type=USR  
fileset=fset0  
blockQuota=25G  
blockLimit=30G  
filesQuota=10K  
filesLimit=11K
```

Then issue the command:

```
# mmsetquota -F /tmp/defaultQuotaExample
```

4. To activate quota checking, use the **mmquotaon** command.
5. To list quotas, use the **mmlsquota** command:

The default quotas can be deactivated by issuing the **mmdefquotaoff** command.

For fileset recommendations, see *Filesets and quotas* in *IBM Spectrum Scale: Advanced Administration Guide*.

For complete usage information, see “mmchfs command” on page 371, “mmcrfs command” on page 414, “mmdefquota command” on page 434, “mmdefquotaoff command” on page 437, “mmdefquotaon command” on page 440, “mmedquota command” on page 481, “mmlsfs command” on page 536, and “mmsetquota command” on page 657.

Implications of quotas for different protocols

Quotas can mean different things for different protocols. This section describes how quotas affect the SMB and NFS protocols.

Quotas are stored and enforced in the file system. See Chapter 8, “Managing GPFS quotas,” on page 169 for details on how to enable and use quotas.

- SMB protocol and quotas

For SMB clients, quotas can limit the used and free space reported to clients:

- If the SMB option **gpfs:dfreequota** is set, the user quota for the current user and the group quota for the user's primary group are queried during the free space query:
 - If the block limit is reached, the free space is reported as 0 and the size of the share is reported with the currently used data.
 - If the soft block quota is exceeded for longer than the block grace time, the free space is reported as 0 and the size of the share is reported with the currently used data.
 - If no limit is exceeded, the free space is reported as the free space according to the lowest quota limit.

- If no quota is in place, the size and free space as queried from the underlying file system are reported.
- In the case of per-fileset user and group quotas, the quotas are only queried from the root folder of the export. If a subdirectory inside the share is in a different fileset, the user and group quotas are not considered for the free space report.

Note: For including fileset quotas in the reported free space, configure the underlying file system with the `--filesetdf` flag (in `mmcrfs` or `mmchfs`). It is not possible to query or change individual quotas from a SMB client system.

- NFS protocol and quotas

It is not possible to query or change individual quotas from a NFS client system. User and group quotas are not included in the reported free space to a client. To include fileset quotas in the reported space to a client, configure the underlying file system with the `--filesetdf` flag (in `mmcrfs` or `mmchfs`).

- Object protocol and quotas:

- There are two applicable levels of quotas: The quotas in the file system (see Chapter 8, “Managing GPFS quotas,” on page 169) and the quotas managed by Swift.
- Swift quotas can be used as account and container quotas. The quota values are set as account and container metadata entries. For more information, see the OpenStack Swift documentation.
- When using file system quotas, it is important to consider that all objects stored by Swift are stored with the same owner and owning group (`swift:swift`).

Explicitly establishing and changing quotas

Use the `mmedquota` command to explicitly establish or change file system quota limits for users, groups of users, or filesets.

When setting quota limits for a file system, replication within the file system should be considered. See “Listing quotas” on page 177.

The `mmedquota` command opens a session using your default editor, and prompts you for soft and hard limits for blocks and inodes. For example, to set user quotas for user `jesmith`, enter:

```
mmedquota -u jesmith
```

The system displays information in your default editor similar to:

```
*** Edit quota limits for USR jesmith:
NOTE: block limits will be rounded up to the next multiple block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs0: blocks in use: 24576K, limits (soft = 0K, hard = 0K)
      inodes in use: 0, limits (soft = 0K, hard = 0K)
```

Note: A quota limit of zero indicates **no** quota limits have been established.

The current (in use) block and inode usage is for display only; it cannot be changed. When establishing a new quota, zeros appear as limits. Replace the zeros, or old values if you are changing existing limits, with values based on the user's needs and the resources available. When you close the editor, GPFS checks the values and applies them. If an invalid value is specified, GPFS generates an error message. If this occurs, reenter the `mmedquota` command. If the scope of quota limit enforcement is the entire file system, `mmedquota` will list all instances of the same user (for example, `jesmith`) on different GPFS file systems; if the quota enforcement is on a per-fileset basis, `mmedquota` will list all instances of the same user on different filesets on different GPFS file systems.

You may find it helpful to maintain a *quota prototype*, a set of limits that you can apply by name to any user, group, or fileset without entering the individual values manually. This makes it easy to set the same

limits for all. The **mmedquota** command includes the **-p** option for naming a prototypical user, group, or fileset on which limits are to be based. The **-p** flag can only be used to propagate quotas from filesets within the same file system.

For example, to set group quotas for all users in a group named **blueteam** to the prototypical values established for **prototeam**, issue:

```
mmedquota -g -p prototeam blueteam
```

You may also reestablish default quotas for a specified user, group of users, or fileset when using the **-d** option on the **mmedquota** command.

Note: You can use the **mmsetquota** command as an alternative to the **mmedquota** command.

For complete usage information, see “mmedquota command” on page 481 and “mmsetquota command” on page 657.

Setting quotas for users on a per-project basis

A file system must be properly configured in order to set quotas for users. Use this information to set quotas for any number of users on a per-project basis across protocols.

1. Ensure the file system is configured correctly to use quotas:
 - a. The **-Q yes** option must be in effect for the file system.
 - b. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect.

Note: If **--perfileset-quota** is in effect, all users and groups in the fileset **root** will not be impacted by default quota unless they are explicitly set.

The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options.

Here are some examples:

- a. If a GPFS cluster is created with configuration profile file, `example.profile`, which contains the following lines:

```
%filesystem
quotasAccountingEnabled=yes
quotasEnforced=user;group;fileset
perfilesetQuotas=yes
```

then, when a file system is created, those quota attributes will be set automatically: quota accounting will be enabled on a perfileset basis for users and groups, and quotas will automatically be enforced. This means that when a quota is reached, the end user will not be able to add more data to the file system.

mmcrfs fs5 nsd8

A listing of the file system config, using the **mmlsfs** command, will show the following attributes and values, having been set by the **mmcrfs** command:

mmlsfs fs5

```
...
-Q                user;group;fileset      Quotas accounting enabled
                  user;group;fileset      Quotas enforced
                  none                    Default quotas enabled
--perfileset-quota Yes                    Per-fileset quota enforcement
....
```

For more information on **mmcrcluster** user-defined profiles, see “mmcrcluster command” on page 403.

- b. Whether or not a GPFS was created with a configuration profile file, a GPFS file system can be created with the quota attributes to be set. This can be done by calling the configuration profile file explicitly from the command line:

```
mmcrfs fs6 nsd9 --profile=example
```

For more information on user-defined profiles, see “mmcrfs command” on page 414.

2. Create a fileset on the file system for the project using the **mmcrfileset** command.

For example:

```
mmcrfileset fs5 projectX --inode-space=new
```

Note: It is recommended to create an independent fileset for the project.

3. Link the fileset using the **mmlinkfileset** command.

The file system, fs5, must be mounted, using the **mmmout** command. For example:

```
mmmout fs5 -a
```

```
mmchfs fs5 --inode-limit 400000:300000
```

Output:

```
Set maxInodes for inode space 0 to 400000
Fileset root changed.
```

```
mmlinkfileset fs5 projectX -J /gpfs/fs5/projectX
```

4. Create export/share using the newly created fileset as the export/share path. For more information, see the “mmnfs command” on page 570 and the “mmsmb command” on page 663. For example:

```
mmnfs export add /gpfs/fs5/projectX
```

5. If needed, specify absolute fileset inode limits using the **mmchfileset** command. A fileset inode limit is analogous to saying this is how many files and directories the project is likely to produce. This is not something that can be easily recommended. Nonetheless, here is an example of how that can be set and listed for the file system and fileset:

```
mmchfileset fs5 projectX --inode-limit 200000
```

Output:

```
Set maxInodes for inode space 1 to 200000
Fileset projectX changed.
```

```
mmlsfileset fs5 -L
```

Output:

```
Filesets in file
system 'fs5':
  Name      Id  RootInode  ParentId  Created                Inode Space  MaxInodes  AllocInodes  Comment
  root      0   3          --        Sat Mar 28 13:40:33 2015  0 400000    310656      root        fileset
  projectX  1   524291    --        Sat Mar 28 14:54:13 2015  1 200000    100032
```

6. Now that there is a fileset limit in place, which is entirely optional, to set group quota limits on the project, that is, on fileset **projectX** on file system **fs5**, use the **mmsetquota** command. For example, if the group **groupY** will access **projectX**:

```
mmsetquota fs5:projectX --group groupY --block 128G --files 150K
```

Here, the perfileset quota needs to be enabled on **fs5** as in step 1, and the group **groupY** must have a GID (group ID) on the GPFS cluster. The **block** parameter is used to specify the maximum size of the data on the storage device and the **files** parameter is used to specify the maximum number for files (or directories) the **groupY** is able to consume or create on **projectX**, a fileset of file system **fs5** that is exported through NFS in this example.

At this point, the quota accounting needs to be refreshed on the file system using the **mmcheckquota** command, and then a reporting of the quota limits on **projectX** can take place using the **mmrepquota** command. For example:

```
mmcheckquota fs5
```

```
mmrepquota fs5:projectX
```

Output:

Name	fileset	type	Block Limits					File Limits				
			KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace
root	projectX	USR	0	0	0	0	none	1	0	0	0	none
root	projectX	GRP	0	0	0	0	none	1	0	0	0	none
groupY	projectX	GRP	0	134217728	0	0	none	0	153600	0	0	none

7. If the project grows, or shrinks, and quota changes at the group level are needed, the **mmsetquota** command can again be used to change the quotas for **groupY** on **projectX**. For example, if the expected limits for **projectX** doubles:

```
mmsetquota fs5:projectX --group groupY --block 256G --files 300K
```

```
mmrepquota fs5:projectX
```

Output:

Name	fileset	type	Block Limits					File Limits				
			KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace
root	projectX	USR	0	0	0	0	none	1	0	0	0	none
root	projectX	GRP	0	0	0	0	none	1	0	0	0	none
groupY	projectX	GRP	0	268435456	0	0	none	0	307200	0	0	none

8. If the project is projected to exceed the inode limits for the fileset and file system, then these can also be adjusted upwards. For more information, see “**mmchfs** command” on page 371.

Checking quotas

The **mmcheckquota** command counts inode and space usage for a file system and writes the collected data into quota files.

You must use the **mmcheckquota** command if any of the following are true:

1. Quota information is lost due to node failure.

Node failure could leave users unable to open files or deny them disk space that their quotas should allow.

2. The *in doubt* value approaches the quota limit. To see the *in doubt* value, use the **mmlsquota** or **mmrepquota** commands.

As the sum of the *in doubt* value and the current usage may not exceed the hard limit, the actual block space and number of files available to the user, group, or fileset may be constrained by the *in doubt* value. Should the *in doubt* value approach a significant percentage of the quota, use the **mmcheckquota** command to account for the lost space and files.

Note: Running **mmcheckquota** is also recommended (in an appropriate time slot) if the following message is output by **mmrepquota**, **mmlsquota**, or **mmedquota**:

Quota accounting information is inaccurate and quotacheck must be run.

When issuing the **mmcheckquota** command on a mounted file system, negative *in doubt* values may be reported if the quota server processes a combination of up-to-date and back-level information. This is a transient situation and may be ignored.

During the normal operation of file systems with quotas enabled (not running **mmcheckquota** online), the usage data reflects the actual usage of the blocks and inodes in the sense that if you delete files you should see the usage amount decrease. The *in doubt* value does not reflect how much the user has used already, it is just the amount of quotas that the quota server has assigned to its clients. The quota server does not know whether the assigned amount has been used or not. The only situation where the *in doubt* value is important to the user is when the sum of the usage and the *in doubt* value is greater than the user's quota hard limit. In this case, the user is not allowed to allocate more blocks or inodes unless he brings the usage down.

For example, to check quotas for the file system **fs1** and report differences between calculated and recorded disk quotas, enter:

```
mmcheckquota -v fs1
```

The information displayed shows that the quota information for **USR7** was corrected. Due to a system failure, this information was lost at the server, which recorded 0 subblocks and 0 files. The current usage data counted is 96 subblocks and 3 files. This is used to update the quota:

```
fs1: quota check found the following differences:  
USR7: 96 subblocks counted (was 0); 3 inodes counted (was 0)
```

Note: In cases where small files do not have an additional block allocated for them, quota usage may show less space usage than expected.

For complete usage information, see the “mmcheckquota command” on page 361.

Listing quotas

The **mmlsquota** command displays the file system quota limits, default quota limits, and current usage information.

If the scope of quota limit enforcement is the entire file system, **mmlsquota -u** or **mmlsquota -g** will list all instances of the same user or group on different GPFS file systems. If the quota enforcement is on a per-fileset basis, **mmlsquota -u** or **mmlsquota -g** will list all instances of the same user or group on different filesets on different GPFS file systems.

GPFS quota management takes replication into account when reporting on and determining whether quota limits have been exceeded for both block and file usage. If either data or metadata replication is enabled, the values reported by both the **mmlsquota** command and the **mmrepquota** command may exceed the corresponding values reported by commands like **ls**, **du**, and so on. The difference depends on the level of replication and on the number of replicated file system objects. For example, if data block replication is set to 2, and if all files are replicated, then the reported block usage by the **mmlsquota** and **mmrepquota** commands will be double the usage reported by the **ls** command.

When the **mmlsquota** command is issued, negative *in doubt* values may be reported if the quota server processes a combination of up-to-date and back-level information. This is a transient situation and may be ignored.

Display the quota information for one user, group of users, or fileset with the **mmlsquota** command. If none of the options **-g**, **-u**, or **-j** are specified, the default is to display only user quotas for the user who issues the command.

To display default quota information, use the **-d** option with the **mmlsquota** command. For example, to display default quota information for users of all the file systems in the cluster, issue this command:

```
mmlsquota -d -u
```

The system displays information similar to:

Default Block Limits(KB)				Default File Limits		
Filesystem type	quota	limit		quota	limit	Remarks
fs1	USR	5242880	6291456	0	0	

Default Block Limits(KB)				Default File Limits		
Filesystem type	quota	limit		quota	limit	Remarks
fs2	USR	no default limits				

In this example, file system **fs1** shows that the default block quota for users is set at 5 GB for the soft limit and 6 GB for the hard limit. For file system **fs2**, no default quotas for users have been established.

When **mmquota -d** is specified in combination with the **-u**, **-g**, or **-j** options, default file system quotas are displayed. When **mmquota -d** is specified without any of the **-u**, **-g**, or **-j** options, default fileset-level quotas are displayed.

If you issue the **mmquota** command with the **-e** option, the quota system collects updated information from all nodes before returning output. If the node to which *in-doubt* space was allocated should fail before updating the quota system about its actual usage, this space might be lost. Should the amount of space in doubt approach a significant percentage of the quota, run the **mmcheckquota** command to account for the lost space.

To collect and display updated quota information about a group named **blueteam**, specify the **-g** and **-e** options:

```
mmquota -g blueteam -e
```

The system displays information similar to:

Filesystem type	Block Limits					File Limits					
	KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace	
Disk quotas for group blueteam (gid 100):											
fs1	GRP	45730	52000	99000	1335	none	411	580	990	19	none

For complete usage information, see “mmquota command” on page 559.

Activating quota limit checking

Quota limit checking can be activated for users, groups, filesets, or any combination of these three.

You can have quotas activated automatically whenever the file system is mounted by specifying the quota option (**-Q yes**) when creating (**mmcrfs -Q yes**) or changing (**mmchfs -Q yes**) a GPFS file system. When creating a file system, the default is to **not** have quotas activated, so you must specify this option if you want quotas activated.

The **mmquotaon** command is used to turn quota limit checking back on if it had been deactivated by issuing the **mmquotaoff** command. Specify the file system name, and whether user, group, or fileset quotas are to be activated. If you want all three fileset quotas activated (user, group, and fileset), specify only the file system name. After quotas have been turned back on, issue the **mmcheckquota** command to count inode and space usage.

For example, to activate user quotas on the file system **fs1**, enter:

```
mmquotaon -u fs1
```

To confirm the change, enter:

```
mmfsfs fs1 -Q
```

The system displays output similar to:

flag	value	description
-Q	user	Quotas enforced

For complete usage information, see “mmquotaon command” on page 619 and “mmfsfs command” on page 536.

Deactivating quota limit checking

During normal operation, there is no need to deactivate quota enforcement. The only reason you might have to deactivate quota enforcement is when users are denied allocation that their quotas should allow, due to loss of quota information during node failure.

If this occurs, use the **mmcheckquota** command after reactivating quotas to reconcile allocation data. When quota enforcement is deactivated, disk space and file allocations are made without regard to limits.

The **mmquotaoff** command is used to deactivate quota limit checking. Specify the file system name and whether user, group, or fileset quotas, or any combination of these three, are to be deactivated. If you want all types of quotas deactivated, specify only the file system name.

For example, to deactivate only user quotas on the file system **fs1**, enter:

```
mmquotaoff -u fs1
```

To confirm the change, enter:

```
mmfsfs fs1 -Q
```

The system displays output similar to:

```
flag value      description
-----
-Q group;fileset Quotas enforced
```

For complete usage information, see “mmquotaoff command” on page 617 and “mmfsfs command” on page 536.

Changing the scope of quota limit checking

The scope of quota enforcement is established when quotas are activated. By default, user and group quota limits are enforced across the entire file system. Optionally, the scope of quota enforcement can be limited to an individual fileset boundaries.

The scope of quota enforcement can be changed using the **mmchfs** command and specifying either the **--perfileset-quota** or **--noperfileset-quota** option as needed.

After changing the scope of quota enforcement, **mmcheckquota** must be run to properly update the quota usage information.

Creating file system quota reports

You can have GPFS prepare a quota report for a file system using the **mmrepquota** command.

The quota report lists:

1. Number of files used
2. Amount of disk space used
3. Current quota limits
4. In doubt quotas (disk space allocated but currently unaccounted for)
5. Grace period allowance to exceed the soft limit
6. Whether the quotas have been explicitly set (**e**), are default values at the file system level (**d_fsfs**), are default values at the fileset level (**d_fset**), or initial values (**i**)

The entry type also indicates whether or not default quotas are enabled for the file system (**default on** or **default off**).

Specify whether you want to list only user quota information (**-u** flag), group quota information (**-g** flag), or fileset quota information (**-j** flag) in the **mmrepquota** command. The default is to summarize all three quotas. If the **-e** flag is not specified, there is the potential to display negative usage values as the quota server may process a combination of up-to-date and back-level information. See “Listing quotas” on page 177.

If the scope of quota limit enforcement is the entire file system, **mmrepquota -u** or **mmrepquota -g** will list all users or groups on different GPFS file systems. If the quota enforcement is on a per-fileset basis, **mmrepquota -u** or **mmrepquota -g** will list all instances of the same user or group on different filesets on different GPFS file systems.

To list the group quotas (**-g** option) for all file systems in the cluster (**-a** option), and print a report with header lines (**-v** option), enter:

```
mmrepquota -g -v -a
```

The system displays information similar to:

```
*** Report for GRP quotas on fs1
```

Name	type	Block Limits					File Limits					entryType	
		KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace		
system	GRP	25088	0	0	209120	none	32	0	0	1078	none	default	on
usr	GRP	435256	0	0	199712	none	11	0	0	899	none	d_fsys	

For complete usage information, see “mmrepquota command” on page 627.

Restoring quota files

The method that is used for restoring GPFS quota files depends on the version of GPFS.

The three scenarios for restoring GPFS quota files follow.

1. The file system version is lower than 4.1.0.0.

In scenario 1, quota files can be backed up directly by copying visible quota files and then restored using the **mmcheckquota** command. The newly-specified backup quota files are transferred from normal files to quota files (metadata). Old quota files are converted from metadata to "normal" files, so these old quota files can be deleted.

2. The file system version is 4.1.0.0 or higher, but lower than 4.1.1.0.

In scenario 2, quota files cannot be restored using the **mmcheckquota** command.

3. The file system version is 4.1.1.0 (or higher).

In scenario 3, quota files can be restored using the **mmcheckquota** command. Use the **mmcheckquota --backup** command to back up quota files. You can restore quota files from the former backup quota files. The **mmcheckquota** command copies data from specified backup quota files to "invisible" quota files. You cannot view or delete the original quota files. You can delete specified backup quota files only.

Additional details about the three scenarios for restoring GPFS quota files follow.

In scenarios 1 and 3:

- User, group, and fileset quota files can be restored from a backup copy of the original quota file. When restoring quota files, the backup file must be in the root directory of the GPFS file system.

In scenario 1, if a backup copy of the original quota file does not exist, an empty file will be created when the **mmcheckquota** command is issued.

In scenario 3, the **mmcheckquota** command does nothing and prints an error.

- The user, group, or fileset files can be restored from backup copies by issuing the **mmcheckquota** command with the appropriate options.

1. To restore the user quota file for the file system **fs1** from the backup file **userQuotaInfo**, enter:

```
mmcheckquota -u userQuotaInfo fs1
```

This command must be run offline (that is, no nodes are mounted).

2. This will restore the user quota limits set for the file system, but the usage information will not be current. To bring the usage information to current values, the command must be reissued:

```
mmcheckquota fs1
```

In scenario 1, if you want to nullify all quota configuration and then reinitialize it, follow these steps:

1. Remove the existing quota files that are corrupted.
 - a. Disable quota management:

```
mmchfs fs1 -Q no
```
 - b. Remove the **user.quota**, **group.quota**, and **fileset.quota** files.
2. Enable quota management.
 - a. Issue the following command:

```
mmchfs fs1 -Q yes
```
3. Reestablish quota limits by issuing the **mmedquota** command or the **mmdefedquota** command.
4. Gather the current quota usage values by issuing the **mmcheckquota** command.

In scenario 2, quota files do not exist externally. Therefore, use **mmbackupconfig** and **mmrestoreconfig** to restore quota configurations.

For complete usage information, see “mmcheckquota command” on page 361, “mmdefedquota command” on page 434 and “mmedquota command” on page 481.

Chapter 9. Managing GUI administrators

GUI administrators of the IBM Spectrum Scale system can monitor, configure, and manage the IBM Spectrum Scale system and are distinguished from the data users.

You can manage GUI users either locally within the system or in an external authentication server such as Microsoft Active Directory (AD) or Lightweight Directory Access Protocol Server (LDAP). By default, the IBM Spectrum Scale system uses an internal authentication repository for GUI users. To use an external AD or LDAP server, you need to disable the internal user repository that is used for the GUI user management and enable the LDAP/AD repository.

Managing administrative users locally in the IBM Spectrum Scale system

You can create users who can perform different administrative tasks on the system. Each user must be part of a user group or multiple groups that are defined on the system. When you create a new user, you assign the user to one of the default user groups or to a custom user group. User groups are assigned with predefined roles that authorize the users within that group to a specific set of operations on the GUI.

Use the **Access > GUI Users** page to create users and add them to a user group.

Predefined roles are assigned to user groups to define the working scope within the GUI. If a user is assigned to more than one user group, the permissions are additive, not restrictive. The predefined role names cannot be changed.

The following are the default user groups:

- **Administrator**
Manages all functions on the system except those deals with managing users, user groups, and authentication.
- **SecurityAdmin**
Manages all functions on the system, including managing users, user groups, and user authentication.
- **SystemAdmin**
Manages clusters, nodes, alert logs, and authentication.
- **StorageAdmin**
Manages disks, file systems, pools, filesets, and ILM policies.
- **SnapAdmin**
Manages snapshots for file systems and filesets.
- **DataAccess**
Controls access to data. For example, managing access control lists.
- **Monitor**
Monitors objects and system configuration but cannot configure, modify, or manage the system or its resources.
- **ProtocolAdmin**
Manages object storage and data export definitions of SMB and NFS protocols.
- **UserAdmin**
Manages access for GUI users. Users who are part of this group have edit permissions only in the Access pages of the GUI.

The IBM Spectrum Scale system is delivered with a default GUI user named *admin*. This user is also stored in the local repository. You can log in to the system by using this user name to create additional GUI users and groups in local user repository.

Use the various controls that are available under the **Password Policy** tab of the GUI Users page to enforce strong passwords for the users. You can modify or expire password of the individual users or all the users that are created in the system. If the password is set as expired, the users will be prompted to change the password in the next login.

Use the various controls that are available under the Password Policy tab of the GUI Users page to enforce strong passwords for the users. You can modify or expire password of the individual users or all the users that are created in the system. If the password is set as expired, the users will be prompted to change the password in the next login.

User groups

Users who are part of Security Administrator and User Administrator user groups can create role-based user groups where any users that are added to the group adopt the role that is assigned to that group.

Roles apply to users on the system and are based on the user group to which the user belongs. A user can be part of multiple user groups so that a single user can play multiple roles in the system. You can assign the following roles to your user groups:

- **Administrator**
Users can access all functions on the GUI except those deals with managing users and user groups.
- **Security Administrator**
Users can access all functions on the GUI, including managing users and user groups.
- **System Administrator**
Users manage clusters, nodes, and alert logs.
- **Storage Administrator**
Users manage disks, file systems, pools, and filesets.
- **Snapshot Administrator**
Users manage snapshots for file systems, filesets.
- **Monitor**
Users can view objects and system configuration but cannot configure, modify, or manage the system or its resources.
- **Data Access**
Users can perform the following tasks:
 - Edit owner, group, and ACL of any file or path through the **Access > File System ACL > Files and Directories** page.
 - Edit owner, group, and ACL for a non-empty directory of a file system, fileset, NFS export, or SMB share.
 - Create or delete object containers through the **Object > Accounts** page.
- **Protocol Administrator**
Users manage object storage and data export definitions of SMB and NFS protocols.
- **User Administrator**
Users manage GUI users and user groups.

Managing GUI administrators in an external authentication server

By default, the IBM Spectrum Scale uses an internal authentication repository for the GUI administrators. To use an external AD or LDAP server to manage the GUI administrators, perform the following steps:

1. Disable the internal user repository by performing the following steps:
 - a. Access the server that is running the GUI and open the following file by using a text editor: `/opt/ibm/wlp/usr/servers/gpfsgui/server.xml`
 - b. Comment out the two elements that are referring to the `FscclUserRepo/Registry` that are highlighted in boldface in the following example:

```

<server description="GSS GUI">
  <featureManager>
    <feature>jsp-2.2</feature>
    <feature>localConnector-1.0</feature>
    <feature>jdbc-4.0</feature>
    <feature>ssl-1.0</feature>
    <feature>servlet-3.0</feature>
    <feature>appSecurity-2.0</feature>
    <!-- <feature>usr:FscclUserRepo</feature> -->
    <feature>jndi-1.0</feature>
  </featureManager>
  <!-- <fscclUserRegistry prefFile="\${server.config.dir}/preferences.xml"/> -->
  [...]

```

2. Add the LDAP or AD feature to the WebSphere® Liberty to enable LDAP or AD support in WebSphere Liberty by performing the following steps:

- a. Access the `server.xml` file, which is at the following location: `/opt/ibm/wlp/usr/servers/gpfsgui/server.xml`
- b. Add the entry `ldapRegistry-3.0` in the `server.xml` as shown in the following example:

```

<server description="GSS GUI">
  <featureManager>
    <feature>jsp-2.2</feature>
    <feature>localConnector-1.0</feature>
    <feature>jdbc-4.0</feature>
    <feature>ssl-1.0</feature>
    <feature>servlet-3.0</feature>
    <feature>appSecurity-2.0</feature>
    <!-- <feature>usr:FscclUserRepo</feature> -->
    <feature>jndi-1.0</feature>
    <feature>ldapRegistry-3.0</feature>
  </featureManager>

```

Note: When the IBM Spectrum Scale system is updated to the latest release, the `server.xml` is overwritten as part of the update. Therefore, if an external authentication server is used for managing GUI administrators, the `server.xml` file must be edited accordingly after every system update.

3. Configure `<ldapRegistry>` element in the LDAP or AD repository. Depending on the type of the external server, the configuration element can have different attribute values. Sample configurations for AD and IBM Directory Server are given in the following example:

Active Directory Server

```

<ldapRegistry id="ldap"
  host="ldapservers.mycity.mycompany.com" port="389" ignoreCase="true"
  baseDN="cn=users,dc=adtest,dc=mycity,dc=mycompany,dc=com"
  bindDN="cn=testuser,cn=users,dc=adtest,dc=mycity,dc=mycompany,dc=com"
  bindPassword="testuserpwd"
  ldapType="Microsoft Active Directory"
  sslEnabled="false">
  <activatedFilters
    userFilter="(&(sAMAccountName=%v)(objectcategory=person))"
    groupFilter="(&(cn=%v)(objectcategory=group))"
    userIdMap="user:sAMAccountName"
    groupIdMap="*:cn"
    groupMemberIdMap="memberOf:member">
  </activatedFilters>
</ldapRegistry>

```

IBM Directory Server:

```

<ldapRegistry id="ldap"
  host="ldapserver.mycity.mycompany.com" port="389" ignoreCase="true"
  baseDN="o=mycompany,c=us"
  ldapType="IBM Tivoli Directory Server"
  sslEnabled="false">
<idsFilters
  userFilter="(&(uid=%v)(objectclass=ePerson))"
groupFilter="(&(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfURLs)))"
userIdMap="*:uid"
groupIdMap="*:cn"
groupMemberIdMap="mycompany-allGroups:member;mycompany-allGroups:uniqueMember;
groupOfNames:member;groupOfUniqueNames:uniqueMember">
</idsFilters>
</ldapRegistry>

```

For more information on the advanced configuration options or for enabling SSL, see Configuring LDAP user registries in Liberty.

4. Establish the LDAP group to GUI role mapping. After the GUI server was restarted, you must review the groups to roles mapping and add/remove group to role mappings as necessary.
5. View and modify the existing group to role mappings. You can view the existing groups by using the **lsusergrp** command. Adding and removing groups can be done by using **mkusergrp** and **rmusergrp** respectively.

Note: The commands that are used to manage the GUI administrators are not available in the same path where all other IBM Spectrum Scale commands are located. The GUI user management commands are located at the following location in the system: `/usr/lpp/mmfs/gui/cli`

6. Create a group to role mapping for initial access. For initial GUI access, you need to map one existing LDAP or AD group to the SecurityAdmin GUI role. The group name needs to match the CN attribute of the corresponding group in the external LDAP or AD repository. Log on to the server that is hosting the GUI and run the following command, which maps the specified LDAP group to the GUI role SecurityAdmin.

```
# /usr/lpp/mmfs/gui/cli/mkusergrp mySecurityAdminLDAPGroup --role securityadmin
```

7. After the initial setup, any additional group mappings can be created through the GUI by using the **Create Group Mapping** option that is available in the **Access > GUI Access** page of the IBM Spectrum Scale management GUI.

Note: The GUI Access page is available only if an external authentication server is enabled to manage the GUI user authentication. If an internal user repository is used for GUI user authentication, the GUI displays GUI Users page to create and manage GUI users and user roles.

Chapter 10. Managing GPFS access control lists

Access control protects directories and files by providing a means of specifying who is granted access. GPFS access control lists are either traditional ACLs based on the POSIX model, or NFS V4 ACLs. NFS V4 ACLs are very different than traditional ACLs, and provide much more fine control of file and directory access. A GPFS file system can also be exported using NFS.

Management of GPFS access control lists (ACLs) and NFS export includes these topics:

- “Traditional GPFS ACL administration”
- “NFS V4 ACL administration” on page 191
- “NFS and GPFS” on page 195

Traditional GPFS ACL administration

Support for NFS V4 access control lists (ACLs) has been added to traditional ACL support. NFS V4 ACLs are very different than the traditional ones.

If you are using NFS V4 ACLs, see “NFS V4 ACL administration” on page 191. Both ACL types may coexist in a single GPFS file system.

Traditional GPFS ACLs are based on the POSIX model. Traditional GPFS access control lists (ACLs) extend the base permissions, or standard file access modes, of read (r), write (w), and execute (x) beyond the three categories of file owner, file group, and other users, to allow the definition of additional users and user groups. In addition, GPFS introduces a fourth access mode, control (c), which can be used to govern who can manage the ACL itself.

In this way, a traditional ACL can be created that looks like this:

```
#owner:jesmith
#group:team_A
user::rwx
group::rwx-
other::--x-
mask::rwx
user:alpha:r-xc
group:audit:r-x-
group:system:rwx-
```

In this ACL:

- The first two lines are comments showing the file's owner, **jesmith**, and group name, **team_A**
- The next three lines contain the base permissions for the file. These three entries are the minimum necessary for a GPFS ACL:
 1. The permissions set for the file owner (**user**), **jesmith**
 2. The permissions set for the owner's **group**, **team_A**
 3. The permissions set for **other** groups or users outside the owner's group and not belonging to any named entry
- The next line, with an entry type of **mask**, contains the maximum permissions allowed for any entries other than the owner (the **user** entry) and those covered by **other** in the ACL.
- The last three lines contain additional entries for specific users and groups. These permissions are limited by those specified in the mask entry, but you may specify any number of additional entries up to a memory page (approximately 4 K) in size.

Traditional GPFS ACLs are fully compatible with the base operating system permission set. Any change to the base permissions, using the **chmod** command, for example, modifies the corresponding GPFS ACL as well. Similarly, any change to the GPFS ACL is reflected in the output of commands such as **ls -l**. Note that the control (c) permission is GPFS specific. There is no comparable support in the base operating system commands. As a result, the (c) permission is visible only with the GPFS ACL commands.

Each GPFS file or directory has an *access ACL* that determines its access privileges. These ACLs control who is allowed to read or write at the file or directory level, as well as who is allowed to change the ACL itself.

In addition to an *access ACL*, a directory may also have a *default ACL*. If present, the default ACL is used as a base for the access ACL of every object created in that directory. This allows a user to protect all files in a directory without explicitly setting an ACL for each one.

When a new object is created, and the parent directory has a default ACL, the entries of the default ACL are copied to the new object's access ACL. After that, the base permissions for user, mask (or group if mask is not defined), and other, are changed to their intersection with the corresponding permissions from the mode parameter in the function that creates the object.

If the new object is a directory, its default ACL is set to the default ACL of the parent directory. If the parent directory does not have a default ACL, the initial access ACL of newly created objects consists only of the three required entries (user, group, other). The values of these entries are based on the mode parameter in the function that creates the object and the umask currently in effect for the process.

Administrative tasks associated with traditional GPFS ACLs are:

1. "Setting traditional GPFS access control lists"
2. "Displaying traditional GPFS access control lists" on page 189
3. "Changing traditional GPFS access control lists" on page 190
4. "Deleting traditional GPFS access control lists" on page 190

Setting traditional GPFS access control lists

Use the following information to set GPFS ACLs:

GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Access > File System ACL**.

Use the **mmputacl** command to set the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to set the ACL for a file named **project2.history**, we can create a file named **project2.acl** that contains:

```
user::rwx-
group::rwx-
other::--x-
mask::rwx-
user:alpha:r-xc
group:audit:rw--
group:system:rwx-
```

In this example,

- The first three lines are the required ACL entries setting permissions for the file's owner, the owner's group, and for processes that are not covered by any other ACL entry.
- The last three lines contain named entries for specific users and groups.

- Because the ACL contains named entries for specific users and groups, the fourth line contains the required mask entry, which is applied to all named entries (entries other than the **user** and **other**).

Once you are satisfied that the correct permissions are set in the ACL file, you can apply them to the target file with the **mmputacl** command. For example, to set permissions contained in the file **project2.acl** for the file **project2.history**, enter:

```
mmputacl -i project2.acl project2.history
```

To confirm the changes, enter:

```
mmgetacl project2.history
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwx
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwx #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

Although you can issue the **mmputacl** command without using the **-i** option to specify an ACL input file, and make ACL entries through standard input, you will probably find the **-i** option more useful for avoiding errors when creating a new ACL.

For complete usage information, see the “**mmputacl** command” on page 614 and the “**mmgetacl** command” on page 500.

Displaying traditional GPFS access control lists

Use the **mmgetacl** command to display the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to display the ACL for the file **project2.history**, enter:

```
mmgetacl project2.history
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwx
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwx #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

The first two lines are comments displayed by the **mmgetacl** command, showing the owner and owning group. All entries containing permissions that are not allowed (because they are not set in the mask entry) display with a comment showing their effective permissions.

For complete usage information, see the “**mmgetacl** command” on page 500

Applying an existing traditional GPFS access control list

To apply the same traditional ACLs from one file or directory to another:

1. Issue the **mmgetacl** command with the **-o** option to place the information in an output file.
2. Apply the ACLs to the new file or directory by issuing the **mmputacl** command with the **-i** option.

For example, use the **-o** option to specify a file to which the ACL is written:

```
mmgetacl -o old.acl project2.history
```

Then, to assign the same permissions to another file, **project.notes**, enter:

```
mmputacl -i old.acl project.notes
```

To confirm the changes, enter:

```
mmgetacl project.notes
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwx
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwx #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

For complete usage information, see the “mmgetacl command” on page 500 and the “mmputacl command” on page 614.

Changing traditional GPFS access control lists

Use the **mmeditACL** command to change or create the traditional ACL of a file or directory, or the default ACL of a directory. For example, to interactively edit the ACL for the file **project2.history**, enter:

```
mmeditACL project2.history
```

The current ACL entries are displayed using the default editor, provided that the EDITOR environment variable specifies a complete path name. When the file is saved, the system displays information similar to:

```
mmeditACL: 6027-967 Should the modified ACL be applied? (yes) or (no)
```

After responding **yes**, the ACLs are applied.

For complete usage information, see “mmeditACL command” on page 478.

Deleting traditional GPFS access control lists

Use the **mmdelACL** command to delete the extended entries in a traditional ACL of a file or directory, or the default ACL of a directory. For example, to delete the ACL for the directory **project2**, enter:

```
mmdelACL project2
```

To confirm the deletion, enter:

```
mmgetACL project2
```

The system displays information similar to:

```
#owner:uno
#group:system
user::rwx
group::r-x-
other::--x-
```

You cannot delete the base permissions. These remain in effect after this command is executed.

For complete usage information, see “`mmdeacl` command” on page 446 and “`mmgetacl` command” on page 500.

NFS V4 ACL administration

AIX does not allow a file system to be NFS V4 exported unless it supports NFS V4 ACLs. By contrast, Linux does not allow a file system to be NFS V4 exported unless it supports POSIX ACLs.

This is because NFS V4 Linux servers handle NFS V4 ACLs by translating them into POSIX ACLs. For more information, see “Linux ACLs and extended attributes” on page 882.

Note:

This topic does not refer to the NFS Server function included with CES. For information, see “Authorizing protocol users” on page 200.

With AIX, the file system must be configured to support NFS V4 ACLs (with the `-k all` or `-k nfs4` option of the `mmcrfs` or `mmchfs` command). The default for the `mmcrfs` command is `-k all`.

With Linux, the file system must be configured to support POSIX ACLs (with the `-k all` or `-k posix` option of the `mmcrfs` or `mmchfs` command).

Depending on the value (`posix` | `nfs4` | `all`) of the `-k` parameter, one or both ACL types can be allowed for a given file system. Since ACLs are assigned on a per-file basis, this means that within the same file system one file may have an NFS V4 ACL, while another has a POSIX ACL. The type of ACL can be changed by using the `mmputacl` or `mmeditACL` command to assign a new ACL or by the `mmdelACL` command (causing the permissions to revert to the mode which is in effect a POSIX ACL). At any point in time, only a single ACL can be associated with a file. Access evaluation is done as required by the ACL type associated with the file.

NFS V4 ACLs are represented in a completely different format than traditional ACLs. For detailed information on NFS V4 and its ACLs, refer to *NFS Version 4 Protocol* and other information found in the Network File System Version 4 (nfsv4) section of the IETF Datatracker website (datatracker.ietf.org/wg/nfsv4/documents).

In the case of NFS V4 ACLs, there is no concept of a default ACL. Instead, there is a single ACL and the individual ACL entries can be flagged as being *inherited* (either by files, directories, both, or neither). Consequently, specifying the `-d` flag on the `mmputacl` command for an NFS V4 ACL is an error.

NFS V4 ACL Syntax

An NFS V4 ACL consists of a list of ACL entries. Where traditional ACLs can display one entry per line, the GPFS representation of NFS V4 ACL entries are three lines each, due to the increased number of available permissions beyond the traditional `rwxc`.

The first line has several parts separated by colons (':').

- The first part identifies the user or group.
- The second part displays a `rwxc` translation of the permissions that appear on the subsequent two lines.
- The third part is the ACL type. NFS V4 provides both an *allow* and *deny* type.
 - allow* Means to allow (or permit) those permissions that have been selected with an 'X'.
 - deny* Means to not allow (or deny) those permissions that have been selected with an 'X'.
- The fourth and final part is a list of flags indicating *inheritance*.

Valid flag values are:

DirInherit

Indicates that the ACL entry should be included in the initial ACL for subdirectories created in this directory (as well as the current directory).

FileInherit

Indicates that the ACL entry should be included in the initial ACL for files created in this directory.

Inherited

Indicates that the current ACL entry was derived from inherit entries in an NFS v4 ACL of the parent directory.

InheritOnly

Indicates that the current ACL entry should *not* apply to the directory, but *should* be included in the initial ACL for objects created in this directory.

NoPropagateInherit

Indicates that the ACL entry should be included in the initial ACL for subdirectories created in this directory but not further propagated to subdirectories created below *that* level.

As in traditional ACLs, users and groups are identified by specifying the type and name. For example, **group:staff** or **user:bin**. NFS V4 provides for a set of special names that are not associated with a specific local UID or GID. These special names are identified with the keyword **special** followed by the NFS V4 name. These names are recognized by the fact that they end with the character '@'. For example, **special:owner@** refers to the owner of the file, **special:group@** the owning group, and **special:everyone@** applies to all users.

The next two lines provide a list of the available access permissions that may be *allowed* or *denied*, based on the ACL type specified on the first line. A permission is selected using an 'X'. Permissions that are not specified by the entry should be left marked with '-' (minus sign).

These are examples of NFS V4 ACLs.

1. An ACL entry that explicitly allows **READ**, **EXECUTE** and **READ_ATTR** to the **staff** group on a file is similar to this:

```
group:staff:r-x::allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (X)READ_ATTR (-)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

2. A Directory ACL is similar to this. It may include *inherit* ACL entries that do not apply to the directory itself, but instead become the initial ACL for any objects created within the directory.

```
special:group@:----:deny:DirInherit:InheritOnly
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (X)READ_ATTR (-)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

3. A complete NFS V4 ACL is similar to this:

```
#NFSv4 ACL
#owner:smithj
#group:staff
special:owner@:rwx:allow:FileInherit
(X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (-)WRITE_NAMED

special:owner@:rwx:allow:DirInherit:InheritOnly
(X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

user:smithj:rwx:allow
(X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL (X)READ_ATTR (-)READ_NAMED
(X)DELETE (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

ACL entries DELETE and DELETE_CHILD

The ACL entries **DELETE** and **DELETE_CHILD** require special considerations. The effect of various combinations of the **DELETE** attribute for a file, and the **DELETE_CHILD** attribute for its parent directory, is given in Table 15.

In this table, the columns refer to the ACL entry for a given file, and the rows refer to the ACL entry for its parent directory. The various combinations of these attributes produce one of these results:

Permit

Indicates that GPFS permits removal of a file with the combination of file and parent directory ACL entries specified. (Other permission checking may exist within the operating system as well.)

Deny Indicates that GPFS denies (does not permit) removal of a file with the combination of file and parent directory ACL entries specified.

Removal of a file includes renaming the file, moving the file from one directory to another even if the file name remains the same, and deleting it.

Table 15. Removal of a file with ACL entries **DELETE** and **DELETE_CHILD**

	ACL Allows DELETE	ACL Denies DELETE	DELETE not specified	UNIX mode bits only
ACL Allows DELETE_CHILD	Permit	Permit	Permit	Permit
ACL Denies DELETE_CHILD	Permit	Deny	Deny	Deny
DELETE_CHILD not specified	Permit	Deny	Deny	Deny
UNIX mode bits only - wx permissions allowed	Permit	Permit	Permit	Permit
UNIX mode bits only - no w or no x permissions allowed	Permit	Deny	Deny	Deny

The UNIX mode bits are used in cases where the ACL is not an NFS V4 ACL.

NFS V4 ACL translation

NFS V4 access requires that an NFS V4 ACL be returned to clients whenever the ACL is read. This means that if a traditional GPFS ACL is associated with the file, a translation to NFS V4 ACL format must be performed when the ACL is read by an NFS V4 client. Since this translation has to be done, an option (**-k nfs4**) is provided on the **mmgetacl** and **mmeditacl** commands, so that this translation can be seen locally as well.

It can also be the case that NFS V4 ACLs have been set for some file system objects (directories and individual files) prior to administrator action to revert back to a POSIX-only configuration. Since the NFS V4 access evaluation will no longer be performed, it is desirable for the **mmgetacl** command to return an ACL representative of the evaluation that will now occur (translating NFS V4 ACLs into traditional POSIX style). The **-k posix** option returns the result of this translation.

Users may need to see ACLs in their true form as well as how they are translated for access evaluations. There are four cases:

1. By default, the **mmgetacl** command returns the ACL in a format consistent with the file system setting:
 - If **posix** only, it is shown as a traditional ACL.
 - If **nfs4** only, it is shown as an NFS V4 ACL.
 - If **all** formats are supported, the ACL is returned in its true form.
2. The command **mmgetacl -k nfs4** always produces an NFS V4 ACL.
3. The command **mmgetacl -k posix** always produces a traditional ACL.

4. The command **mmgetacl -k native** always shows the ACL in its true form, regardless of the file system setting.

In general, users should continue to use the **mmgetacl** and **mmeditacl** commands without the **-k** flag, allowing the ACL to be presented in a form appropriate for the file system setting. Since the NFS V4 ACLs are more complicated and therefore harder to construct initially, users that want to assign an NFS V4 ACL should use the command **mmeditacl -k nfs4** to start with a translation of the current ACL, and then make any necessary modifications to the NFS V4 ACL that is returned.

Setting NFS V4 access control lists

There is no option on the **mmputacl** command to identify the type (traditional or NFS V4) of ACL that is to be assigned to a file. Instead, the ACL is assumed to be in the traditional format unless the first line of the ACL is:

```
#NFSv4 ACL
```

The lines that follow the first one are then processed according to the rules of the expected ACL type.

An NFS V4 ACL is similar to this:

```
#NFSv4 ACL
#owner:root
#group:system
special:owner@:rwx:allow
(X)READ/LIST (X)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE (X)READ_ACL (-)READ_ATTR (-)READ_NAMED
(X)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (-)WRITE_NAMED

special:owner@:----:deny
(-)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (-)READ_ATTR (X)READ_NAMED
(-)DELETE (X)DELETE_CHILD (X)CHOWN (-)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (X)WRITE_NAMED

user:guest:r-xc:allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE (X)READ_ACL (-)READ_ATTR (-)READ_NAMED
(X)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

user:guest:----:deny
(-)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (-)READ_ATTR (X)READ_NAMED
(-)DELETE (X)DELETE_CHILD (X)CHOWN (-)EXEC/SEARCH (-)WRITE_ACL (X)WRITE_ATTR (X)WRITE_NAMED
```

This ACL shows four ACL entries (an allow and deny entry for each of **owner@** and **guest**).

In general, constructing NFS V4 ACLs is more complicated than traditional ACLs. Users new to NFS V4 ACLs may find it useful to start with a traditional ACL and allow either **mmgetacl** or **mmeditacl** to provide the NFS V4 translation, using the **-k nfs4** flag as a starting point when creating an ACL for a new file.

Displaying NFS V4 access control lists

The **mmgetacl** command displays an existing ACL regardless of its type (traditional or NFS V4). The format of the ACL that is returned depends on the file system setting (**-k** flag), as well as the format of the actual ACL associated with the file. For details, see “NFS V4 ACL translation” on page 193.

Applying an existing NFS V4 access control list

This function is identical, whether using traditional or NFS V4 ACLs. See “Applying an existing traditional GPFS access control list” on page 189.

Changing NFS V4 access control lists

This function is identical, whether using traditional or NFS V4 ACLs. See “Changing traditional GPFS access control lists” on page 190.

Deleting NFS V4 access control lists

Use the **mmdelacl** command to delete NFS V4 ACLs. Once the ACL has been deleted, permissions revert to the mode bits. If the **mmgetacl** command is then used to display the ACL (**mmgetacl -k native**), it appears as a traditional GPFS ACL.

When assigning an ACL to a file that already has an NFS V4 ACL, there are some NFS rules that must be followed. Specifically, in the case of a directory, there will **not** be two separate (access and default) ACLs, as there are with traditional ACLs. NFS V4 requires a single ACL entity and allows individual ACL entries to be flagged if they are to be inherited. Consequently, **mmputacl -d** is not allowed if the existing ACL was the NFS V4 type, since this attempts to change **only** the default ACL. Likewise **mmputacl** (without the **-d** flag) is not allowed because it attempts to change only the access ACL, leaving the default unchanged. To change such an ACL, use the **mmeditACL** command to change the entire ACL as a unit. Alternatively, use the **mmdelACL** command to remove an NFS V4 ACL, followed by the **mmputacl** command.

Considerations when using GPFS with NFS V4 ACLs

There are several constraints that you need to consider when using GPFS with NFS V4 ACLs. For a complete description of these restrictions, see “GPFS exceptions and limitations to NFS V4 ACLs” on page 882.

NFS and GPFS

GPFS file systems may be exported using the Network File System (NFS) protocol from one or more nodes. After export, normal access to the file system can proceed from GPFS cluster nodes or NFS client nodes.

Note: GPFS on Windows does not provide NFS integration.

See the “Authorizing protocol users” on page 200 topic for information on ACLs for protocol users.

Considerations for the interoperability of a GPFS file system include:

- “Exporting a GPFS file system using NFS”
- “NFS usage of GPFS cache” on page 198
- “Synchronous writing using NFS” on page 199
- “Unmounting a file system after NFS export” on page 199
- “NFS automount considerations” on page 199
- “Clustered NFS and GPFS on Linux” on page 199

Note: None of these sections take into account the NFS server integration that is introduced with CES. The integrated NFS server interactions, with the following documented sections, will be addressed in a future release.

Exporting a GPFS file system using NFS

To export a GPFS file system:

1. Create and mount the GPFS file system. In the examples, we assume a file system with a local mount point of **/gpfs**.

For performance reasons, some NFS implementations cache file information on the client. Some of the information (for example, file state information such as file size and timestamp) is not kept up-to-date in this cache. The client may view stale inode data (on **ls -l**, for example) if exporting a GPFS file system with NFS.

If this is not acceptable for a given installation, caching can be turned off by mounting the file system on the client using the appropriate operating system mount option (for example, `-o noac` on Linux NFS clients). Turning off NFS caching results in extra file systems operations to GPFS, and negatively affect its performance.

2. Make sure that the clocks of all nodes in the GPFS cluster are synchronized. If this is not done, NFS access to the data, as well as other GPFS file system operations, may be disrupted.

NFS relies on metadata timestamps to validate the local operating system cache. If the same directory is either NFS-exported from more than one node, or is accessed with both the NFS and GPFS mount point, it is critical that clocks on all nodes that access the file system (GPFS nodes and NFS clients) are constantly synchronized using appropriate software (for example, NTP). Failure to do so may result in stale information seen on the NFS clients.

3. Ensure that NFS is properly configured and running.

For Linux nodes, information on configuring NFS can be obtained at the linuxdoc.org website (www.linuxdoc.org).

For AIX nodes, refer to AIX in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/ssw_aix/welcome) for information about configuring NFS.

4. Use the `mmnfs export add`, `mmnfs export change`, or `mmnfs export remove` command as required.

Note: For IBM Spectrum Scale exports, you must use `mmnfs export add`, `mmnfs export change`, or `mmnfs export remove` command. Editing the `/etc/exports` file to specify file systems to be exported does not work for IBM Spectrum Scale exports; this can only be used for exports that are not IBM Spectrum Scale exports.

Export considerations

Keep these points in mind when exporting a GPFS file system to NFS. The operating system being used and the version of NFS might require special handling or consideration.

Linux export considerations: Linux does not allow a file system to be NFS V4 exported unless it supports POSIX ACLs. For more information, see “Linux ACLs and extended attributes” on page 882.

For Linux nodes only, issue the `exportfs -ra` command to initiate a reread of the `/etc/exports` file.

Starting with Linux kernel version 2.6, an `fsid` value must be specified for each GPFS file system that is exported on NFS. For example, the format of the entry in `/etc/exports` for the GPFS directory `/gpfs/dir1` might look like this:

```
/gpfs/dir1 cluster1(rw,fsid=745)
```

The administrator must assign `fsid` values subject to the following conditions:

1. The values must be unique for each file system.
2. The values must not change after reboots. The file system should be unexported before any change is made to an already assigned `fsid`.
3. Entries in the `/etc/exports` file are not necessarily file system roots. You can export multiple directories within a file system. In the case of different directories of the same file system, the `fsids` must be different. For example, in the GPFS file system `/gpfs`, if two directories are exported (`dir1` and `dir2`), the entries might look like this:

```
/gpfs/dir1 cluster1(rw,fsid=745)  
/gpfs/dir2 cluster1(rw,fsid=746)
```

4. If a GPFS file system is exported from multiple nodes, the `fsids` should be the same on all nodes.

Configuring the directories for export with NFSv4 differs slightly from the previous NFS versions. To configure the directories, do the following:

1. Define the root of the overall exported file system (also referred to as the pseudo root file system) and the pseudo file system tree. For example, to define `/export` as the pseudo root and export `/gpfs/dir1` and `/gpfs/dir2` which are not below `/export`, run:

```
mkdir -m 777 /export /export/dir1 /export/dir2
mount --bind /gpfs/dir1 /export/dir1
mount --bind /gpfs/dir2 /export/dir2
```

In this example, `/gpfs/dir1` and `/gpfs/dir2` are bound to a new name under the pseudo root using the `bind` option of the `mount` command. These bind mount points should be explicitly unmounted after GPFS is stopped and bind-mounted again after GPFS is started. To unmount, use the `umount` command. In the preceding example, run:

```
umount /export/dir1; umount /export/dir2
```

2. Edit the `/etc/exports` file. There must be one line for the pseudo root with `fsid=0`. For the preceding example:

```
/export cluster1(rw,fsid=0)
/export/dir1 cluster1(rw,fsid=745)
/export/dir2 cluster1(rw,fsid=746)
```

The two exported directories (with their newly bound paths) are entered into the `/etc/exports` file.

Large installations with hundreds of compute nodes and a few login nodes or NFS-exporting nodes require tuning of the GPFS parameters `maxFilesToCache` and `maxStatCache` with the `mmchconfig` command.

The general suggestion is for the compute nodes to set `maxFilesToCache` to about 200. The login or NFS nodes should set this parameter much higher, with `maxFilesToCache` set to 1000 and `maxStatCache` set to 50000.

Note: The stat cache is not effective on the Linux platform. Therefore, you need to set the `maxStatCache` attribute to a smaller value, such as 512, and the `maxFilesToCache` attribute to 50000.

This tuning is required for the GPFS token manager (file locking), which can handle approximately 1,000,000 files in memory. The token manager keeps track of a total number of tokens, which equals `5000 * number of nodes`. This will exceed the memory limit of the token manager on large configurations. By default, each node holds 5000 tokens:

- If the user does not specify values for `maxFilesToCache` and `maxStatCache`, the default value of `maxFilesToCache` is 4000, and the default value of `maxStatCache` is 1000.
- On upgrades to GPFS 4.1 from GPFS 3.4 or earlier, the existing defaults (1000 for `maxFilesToCache` and 4000 for `maxStatCache`) remain in effect.

If the user specifies a value for `maxFilesToCache` but does not specify a value for `maxStatCache`, the default value of `maxStatCache` changes to `4*maxFilesToCache`.

If you are running at SLES 9 SP 1, the kernel defines the `sysctl` variable `fs.nfs.use_underlying_lock_ops`, which determines whether the NFS `lockd` is to consult the file system when granting advisory byte-range locks. For distributed file systems like GPFS, this must be set to `true` (the default is `false`).

You can query the current setting by issuing the command:

```
sysctl fs.nfs.use_underlying_lock_ops
```

Alternatively, the `fs.nfs.use_underlying_lock_ops = 1` record can be added to `/etc/sysctl.conf`. This record must be applied after initially booting the node, and after each reboot, by issuing the command:

```
sysctl -p
```

Because the `fs.nfs.use_underlying_lock_ops` variable is currently not available in SLES 9 SP 2 or later, when NFS-exporting a GPFS file system, ensure that your NFS server nodes are at the SP 1 level (unless this variable is made available in later service packs).

For additional considerations when NFS exporting your GPFS file system, refer to *File system creation considerations* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

AIX export considerations: AIX does not allow a file system to be exported by NFS V4 unless it supports NFS V4 ACLs.

NFS export considerations for versions prior to NFS V4: For NFS exported file systems, the version of NFS you are running with may have an impact on the number of inodes you need to cache, as set by both the `maxStatCache` and `maxFilesToCache` parameters on the `mmchconfig` command. The implementation of the `ls` command differs from NFS V2 to NFS V3. The performance of the `ls` command in NFS V3 in part depends on the caching ability of the underlying file system. Setting the cache large enough will prevent rereading inodes to complete an `ls` command, but will put more of a CPU load on the token manager.

Also, the clocks of all nodes in your GPFS cluster must be synchronized. If this is not done, NFS access to the data, as well as other GPFS file system operations, may be disrupted.

NFS V4 export considerations: For information on NFS V4, refer to *NFS Version 4 Protocol* and other information found in the Network File System Version 4 (nfsv4) section of the IETF Datatracker website (datatracker.ietf.org/wg/nfsv4/documents).

To export a GPFS file system using NFS V4, there are two file system settings that must be in effect. These attributes can be queried using the `mmlsfs` command, and set using the `mmcrfs` and `mmchfs` commands.

1. The `-D nfs4` flag is required. Conventional NFS access would not be blocked by concurrent file system reads or writes (this is the POSIX semantic). NFS V4 however, not only allows for its requests to block if conflicting activity is happening, it insists on it. Since this is an NFS V4 specific requirement, it must be set before exporting a file system.

flag value	description
-D nfs4	File locking semantics in effect

2. The `-k nfs4` or `-k all` flag is required. Initially, a file system has the `-k posix` setting, and only traditional GPFS ACLs are allowed. To export a file system using NFS V4, NFS V4 ACLs must be enabled. Since NFS V4 ACLs are vastly different and affect several characteristics of the file system objects (directories and individual files), they must be explicitly enabled. This is done either exclusively, by specifying `-k nfs4`, or by allowing `all` ACL types to be stored.

flag value	description
-k all	ACL semantics in effect

Note: In IBM Spectrum Scale 4.2 release, NFS connections are limited to a maximum of 2250 for a large number of NFS exports. The maximum number of NFS exports supported is 1000.

NFS usage of GPFS cache

Exporting a GPFS file system from a node may result in significant additional demands on the resources at that node. Depending on the number of NFS clients, their demands, and specific mount options, you may want to increase either one or both of the `maxFilesToCache` and `pagepool`. The Ganesha number of open files is set to 1 Mg. By default, `gpfs maxFilesToCache` is set to 4K. If the NFS clients overrun the 4K cache by generating more files, NFS RPC's will fail due to TCP inactivity timeout. Increasing is set to 4K. If the NFS clients overrun the 4K cache by generating more files, NFS RPC's will fail due to TCP inactivity timeout. Increasing `maxFilesToCache` will solve this problem.

See the `mmchconfig` command.

You may also choose to restrict the use of the NFS server node through the normal GPFS path and not use it as either a file system manager node or an NSD server.

Synchronous writing using NFS

With Linux, **w**rite operations are usually asynchronous. If synchronous writes are required over NFS, edit the `/etc/exports` file to include `sync,no_wdelay`.

Unmounting a file system after NFS export

Because NFS use of a GPFS file system may result in a file being held, attempting to unmount a GPFS file system may return a 'Device is busy' error. If this occurs, stop the NFS daemons before attempting to unmount the file system at the NFS server. On Linux, issue this command:

```
/etc/rc.d/init.d/nfs stop
```

On AIX, issue this command:

```
stopsrc -g nfs
```

NFS can be restarted after the unmount completes. On Linux, issue this command:

```
/etc/rc.d/init.d/nfs start
```

On AIX, issue this command:

```
startsrc -g nfs
```

NFS automount considerations

The default file system type when using the automounter daemon is NFS. When the `-fstype` option is not specified, and the server is the local node, a soft-mount of the local directory is done at the desired mount point. JFS is assumed as the only handler of local directories. A GPFS file system local soft-mount does not work implicitly, since the mount request is passed to JFS which then produces an error. When specifying `-fstype mmfs` the local soft-mount works because the mount is then passed to GPFS instead of JFS.

A GPFS soft-mount does not automatically unmount. Setting `-fstype nfs3` causes the local server mounts to always go through NFS. This allows you to have the same `auto.map` file on all nodes whether the server is local or not, and the automatic unmount will occur. If you want local soft-mounts of GPFS file systems while other nodes perform NFS mounts, you should have different `auto.map` files on the different classes of nodes. This should improve performance on the GPFS nodes as they will not have to go through NFS.

Clustered NFS and GPFS on Linux

In addition to the traditional exporting of GPFS file systems using NFS, GPFS allows you to configure a subset of the nodes in the cluster to provide a highly available solution for exporting GPFS file systems via NFS.

The participating nodes are designated as Cluster NFS (CNFS) member nodes and the entire setup is frequently referred to as CNFS or CNFS cluster.

In this solution, all CNFS nodes export the same file systems to the NFS clients. When one of the CNFS nodes fails, the NFS serving load moves from the failing node to another node in the CNFS cluster. Failover is done using recovery groups to help choose the preferred node for takeover.

For more information about CNFS, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Authorizing protocol users

Authorization grants or denies access to resources such as directories, files, commands, and functions. Authorization is applicable to an already authenticated identity, such as an IBM Spectrum Scale data user, an administrative user, or an IBM® service representative. Access to the files and directories of the IBM Spectrum Scale system is managed through access control lists (ACLs). It ensures that only authorized users get access to exports, directories, and files. An access control entry (ACE) is an individual entry in an access control list, and describes the permissions for an individual user or group of users. An ACL can have zero or more ACEs.

Authorizing file protocol users

The IBM Spectrum Scale system uses ACLs to authorize users who access the system through the file protocols such as NFS and SMB.

The GPFS file system supports storing POSIX and NFSv4 ACLs to authorize file protocol users.

SMB service maps the NFSV4 ACL to a security descriptor for SMB clients to form the ACLs. That is, the SMB ACL is derived from the NFSV4 ACL; it is not a separate ACL. Any changes from SMB clients on ACLs are mapped back to the ACLs in the file system.

To get the expected behavior of ACLs, the file system must be configured to use only the NFSV4 ACLs. The default configuration profiles (`/usr/lpp/mmfs/profiles`) that are included with IBM Spectrum Scale contain the required configuration for NFSV4 ACLs in the file system. When manually creating a file system for protocol usage by using the `mmcrfs` command, use the `-k nfs4` option to establish the correct ACL setting. For more details, see the “`mmcrfs` command” on page 414 and the “`mmchfs` command” on page 371.

ACLs can be applied at the following levels:

- Files
- Directories
- Exports

The SMB and NFS protocols allow to manage the ACL permissions. ACLs from both protocols are mapped to the same ACL in the file system. The ACL supports inheritance and you can control the inheritance by using the special inheritance flags.

The export-level ACLs authorize access to NFS and SMB exports. The administrator needs to explicitly make a user or a group of users as the owner of a directory that is being exported as an NFS or SMB export. Export-level ACLs can be changed either through the MMC on a Windows client or through the `sharesec` command.

You can use either `chown` or `chgrp` commands to set an owner for a file or directory. When the export is created with an owner, the ACL management must be done by the owner of the export through the protocol. Additionally, for SMB, privileged users can also perform the ACL management tasks through the protocol. Moreover, by using different security flags, the export-level SMB ACLs can provide more authorization capabilities while creating the SMB export.

For example, after creating an SMB or NFS export with an initial owner, this user can connect to the export by using SMB or NFS protocol to see and manage the ACLs associated with the directory over which the export is created. The export-level SMB ACLs can be changed either through the MMC on a Microsoft Windows client or by using the `sharesec` command. For more information, see the `sharesec` command.

ACLs and POSIX mode bits

The POSIX bits of a file are another authorization method, different from ACLs. POSIX bits can also be used to specify access permissions for a file. You can use the POSIX bits of a file to configure access control for an owner, a group, and for all users to read, update, or run the file. POSIX bits are less flexible than ACLs.

Changing the POSIX modebits also modifies the ACL of an object. When using ACLs for access control, the system administrators might want to ensure that ACLs are not replaced with permissions from POSIX modebits. This behavior can be configured by using the **--allow-permission-change** parameter in **mmcrfileset** and **mmchfileset** commands.

An ACL extends the base permissions or the standard file access modes such as read, write, and execute. ACLs are compatible with UNIX mode bits. Issuing the **chmod** command by the NFS clients overwrite the access privileges that are defined in the ACL by the privileges that are derived from UNIX mode bits. By default, the ACLs are replaced by UNIX mode bits if the **chmod** command is submitted. To allow proper use of ACLs, it is recommended to prevent **chmod** from overwriting the ACLs by setting this parameter to **setAc1only** or **chmodAndSetAc1**.

NFSV3 clients can set and read the POSIX mode bits; NFSV3 clients who set the UNIX permissions modify the ACL to match the UNIX permissions. In most NFS-only cases, the POSIX permissions are used directly. For NFSV3 clients, file sharing with SMB access protection is done by using NFSV4 ACLs but NFSV3 clients can see only the mapping of ACLs to traditional UNIX access permissions. The full NFSV4 ACLs are enforced on the server.

SMB protocol export-level ACLs

Export-level ACLs only apply to SMB exports and they are completely separate from the file system ACLs. The file system ACLs are stored as NFSV4 ACLs. The SMB protocol also has a separate ACL for each export (export-level ACL). That ACL by default grants access to all users. When using export ACLs, users need to have access in the share-level ACL and in the file system ACL to get access to a file.

Export-level ACLs can be changed either through the MMC on a Windows client or through the **sharesec** command. For more information, see the **sharesec** command on any protocol node.

ACL inheritance

The inheritance flags in ACL entry of parent directories are used to control the inheritance of authorization to the child files and directories. The inheritance flag gives you the granularity to specify whether the inheritance defined in an ACL entry applies to the current directory and its children or only to the subdirectories and files that are contained in the parent directory. ACL entries are inherited to the child directories or files at the time of creation. Changes made to the ACL of a parent directory are not propagated to child directories or files. However, in case of SMB, you can specify to propagate the inheritance changes from a parent to all its child by using File Explorer, command line, or PowerShell.

Controlling inheritance of entries inside an ACL

The NFSV4 protocol uses the following flags to specify and control inheritance information of the ACEs:

- *FileInherit*: Indicates that this ACE must be added to each new non-directory file created. This flag is signified by 'f' or `file_inherit`.
- *DirInherit*: Indicates that this ACE must be added to each new directory created. This flag is signified by 'd' or `dir_inherit`.
- *InheritOnly*: Indicates that this ACE is not applied to the parent directory itself, but only inherited by its children. This flag is signified by 'i' or `inherit_only`.
- *NoPropagateInherit*: Indicates that the ACL entry must be included in the initial ACL for subdirectories that are created in this directory but not further propagated to subdirectories created below that level.

In case of SMB, the following list shows how the inheritance flags are linked to the Microsoft Windows inheritance modes:

- This folder only (No bits)
- This folder, subfolder, and files (FileInherit, DirInherit)
- This folder and subfolders (DirInherit)
- This folder and files (FileInherit)
- Subfolders and files only (FileInherit, DirInherit, InheritOnly)
- Subfolders only (DirInherit, InheritOnly)
- Files only (FileInherit, InheritOnly)

ACL best practices

It is essential to properly apply ACLs to the file systems, filesets, and exports, and directories and files to ensure smooth access for the users.

Consider the following points before you create or copy data into the export:

1. Should a group of users be given permission to access the data?
2. Should individual users be given permission to access the data?
3. Should selected users from different groups be given access to selected data?
4. Should the shares be in mixed mode? That is, do you have NFS and SMB clients who access the exports in explicit mode, where the data is accessed either from SMB or NFS?
5. Should the applications that the clients are using over SMB and NFS be given any specific ACL permission?

Setting ACLs for groups

The recommended way to manage access is per group instead of per individual user. This way, users can be easily added to or removed from the group. Providing ACLs to groups has an added advantage of managing inheritance easily for the whole group of users simultaneously. If individual users are added directly to ACLs and you need to make a change, you need to update ACLs of all corresponding directories and files. On the authentication server like Active Directory or LDAP, you can create groups and add users as members and use these groups to give respective access to data.

Setting ACLs for individual users

If you need to set ACLs for individual users where data is created by users in folders that are created by others, it is recommended that you explicitly add the users who need ACLs on that export.

In mixed mode, where the share is used for both NFS and SMB access, parent owner might experience loss of access to the child directory or the files. To avoid such a problem, it is recommended that you provide ACLs explicitly to each user.

Special Owner and Group

The special owner and group dynamically refer to the owner and group of the directory or file that the ACL belongs to. For example, if the owner of a file is changed, all special:owner@ entries in the ACL refers to the new owner. In case of inheritance, this leads to some complexity because those special entries point to the owner and group of the child directory or file that inherits the entry. In many cases, the children do not have the same owner and group as the parent directory. Therefore, the special entries in parent and children refer to different users. This can be avoided by adding static entries (user:'name' or group:'name') to the ACL. These static entries are inherited by name and refer everywhere to the same users. But they are not updated if the owner of the parent is changed. The general recommendation is not to use special:owner@ and special:group@ together with inheritance flags. For more information, see the `mmputacl` command.

Setting ACLs for special IDs

The inheritance of ACL from the owner of a directory to subdirectories and files works only for subdirectories and files that have the same owner as the parent directory. A subdirectory or file that is created by a different owner does not inherit the ACL of a parent directory that is owned by another user.

In case of special access to NFSV4 exports, parent owners might experience loss of access to its child folders and files. To avoid such a problem, for mixed mode, it is recommended that you provide ACLs to groups rather than to individual users.

ACL permissions that are required to work on files and directories

The ACL permissions such as Read Permissions and Read Attributes are required to list a file. A file owner requires only the Read Attributes permission to list a file, since the permission Read Permissions is implied. A different user needs to have both the Read Permissions and Read Attributes permissions enabled to reliably list the file. These permissions are both automatically granted together when read access is granted. If the file is already in the cache of the system because it was listed recently, any user is able to list the file, regardless of the values of the Read Permissions and Read Attributes permission values.

The following table describes the ACL permissions that are required when the user of the file is not the file owner, where "X" denotes permission that is required on file or directory and "P" denotes permission that is required on the parent directory of the file or directory.

Table 16. ACL permissions required to work on files and directories, while using SMB protocol (table 1 of 2)

ACL Operation	ACL Permission					
	Traverse folder / execute file	List folder / read data	Read attribute	Read extended attribute	Create files / write data	Create folders / append data
Execute file	X	X				
List folder		X				
Read data from file		X	X	X		
Read attributes			X			
Create file					X	
Create folder						X
Write data to file		X	X		X	X
Write file attributes						
Write folder attributes						
Delete file		P	X		P	
Delete folder		P	X		P	
Rename file		P	X		P	
Rename folder		P	X		P	P
Read file permissions						
Read folder permissions						
Write file permissions						
Write folder permissions						
Take file ownership						

Table 16. ACL permissions required to work on files and directories, while using SMB protocol (table 1 of 2) (continued)

ACL Operation	ACL Permission					
	Traverse folder / execute file	List folder / read data	Read attribute	Read extended attribute	Create files / write data	Create folders / append data
Take folder ownership						

Table 17. ACL permissions required to work on files and directories, while using SMB protocol (table 2 of 2)

ACL Operation	ACL Permission						
	Write attribute	Write extended attributes	Delete subfolder and files	Delete	Read permissions	Write permissions	Take ownership
Execute file							
List folder							
Read data from file							
Read attributes							
Create file							
Create folder							
Write data to file	X	X					
Write file attributes	X						
Write folder attributes	X						
Delete file			P or X				
Delete folder			P or X				
Rename file			P or X				
Rename folder			P or X				
Read file permissions					X		
Read folder permissions					X		
Write file permissions					X	X	
Write folder permissions					X	X	
Take file ownership							X
Take folder ownership							X

Table 18. ACL permissions required to work on files and directories, while using NFS protocol (table 1 of 2)

ACL Operation	ACL Permission					
	Traverse folder / execute file	List folder / read data	Read attribute	Read extended attribute	Create files / write data	Create folders / append data
Execute file	P, X	X				
List folder	P	X				
Read data from file	P	X				
Read attributes	P					
Create file	P				P	
Create folder	P					P
Write data to file	P				X	X
Write file attributes	P					
Write folder attributes	P					
Delete file	P				P	
Delete folder	P				P	
Rename file	P		X		P	
Rename folder	P		X		P	P
Read file ACL	P					
Read folder ACL	P					
Write file ACL	P					
Write folder ACL	P					
Take file ownership	P					
Take folder ownership	P					

Table 19. ACL permissions required to work on files and directories, while using NFS protocol (table 2 of 2)

ACL Operation	ACL Permission						
	Write attribute	Write extended attributes	Delete subfolder and files	Delete	Read ACL	Write ACL	Take ownership
Execute file							
List folder							
Read data from file							
Read attributes							
Create file							
Create folder							
Write data to file							
Write file attributes							
Write folder attributes							
Delete file			P				
Delete folder			P				
Rename file			P				

Table 19. ACL permissions required to work on files and directories, while using NFS protocol (table 2 of 2) (continued)

ACL Operation	ACL Permission						
	Write attribute	Write extended attributes	Delete subfolder and files	Delete	Read ACL	Write ACL	Take ownership
Rename folder			P				
Read file ACL							
Read folder ACL							
Write file ACL						X	
Write folder ACL						X	
Take file ownership							X
Take folder ownership							X

The following are the considerations on the ACL read and write permissions:

1. For the "Read data from file" operation, the IBM Spectrum Scale system checks the validity of the client requested access mask only if "Read permissions" attribute is enabled on the file. If "Read permissions" attribute is not enabled, then only the "List folder / read data" and "Read attributes" permissions are required to read from the file.
2. For the "Write data to file" operation, the IBM Spectrum Scale system checks the validity of the client requested access mask only if the "Read permissions" attribute is enabled on the file. If the "Read permissions" attribute is not enabled, then only the "Create files / write data" and "Create folders / append data" permissions are required to write to the file.
3. The files that require "Traverse folder / execute file" permission do not require the "Bypass Traverse Check" attribute to be enabled. This attribute is enabled by default on the files.
4. The "Read extended attribute" permission is required by the SMB clients with recent Microsoft Windows versions (for Microsoft Windows 2008, Microsoft Windows 2012, and Microsoft Windows 8 versions) for file copy operations. The default ACLs set without inheritance do not contain this permission. It is recommended that you use inherited permissions where possible and enable this permission in the inherited permissions to prevent the default value to be used and cause problems.

Migrating data through SMB to the IBM Spectrum Scale cluster requires a user ID with the enhanced permissions. The ownership of a file cannot be migrated by a normal IBM Spectrum Scale user. Therefore, you need to configure an "admin user" to allow data migration. For more information on how to configure the "admin users" parameter, see the *mmsmb export add* and *mmsmb export change* sections in the "mmsmb command" on page 663.

Directory traversal permissions that are applicable for SMB ACLs

The following are the considerations on the traverse permissions:

1. It is recommended that you add the "Traverse folder / execute file" permission to all executable files, even if the "Bypass Traverse Check" attribute is enabled on these files. IBM Spectrum Scale checks for the "Traverse folder / execute file" permission on executable files irrespective of the value of the "Bypass Traverse Check" attribute.
2. If the `--cifsbypassTraversalChecking` option is enabled, it allows a user to directly access files and folders that the user owns, and also that are contained under the parent folders for which the user does not have Read or Write permissions. Users without "Read and Execute" access to the share or export in which the user-owned files and folders are located can read and modify the files inside the export for which the user has permissions that are granted by the `--cifsbypassTraversalChecking`

option. However, in this case, operations like rename file and delete file are not granted by default. This is normal SMB behavior. Modify ACLs as required to enable these operations.

For example, in the directory structure /A/B/C, assume that an SMB user has 'read' permission on C but no permissions on A and B. When the `--cifsBypassTraversalChecking` option is set to its default value Yes, this SMB user can access C without having "Traverse Folder" or "Execute File" permissions that are set to allow on A and B, but is still not allowed to browse the content of A and B.

3. The ownership of a file cannot be migrated by a normal user. You must configure and use administrative user credentials to perform data migration. When migrating existing files and directories from other systems to IBM Spectrum Scale, the ACL might not contain explicit traversal rights for the users because the source system can grant this right implicitly. After migrating the files with ACLs, ensure that traversal rights are granted to the parent directory of each exported path.

Working with ACLs

The IBM Spectrum Scale system applies default ACLs for newly created IBM Spectrum Scale file system components such as file system, filesets, file, directories, and exports.

The file system must be created with native ACL type as NFS V4. It is recommended to use the default configuration profiles (`/usr/lpp/mmfs/profiles`) that are included with IBM Spectrum Scale. It contains the required configuration for NFSV4 ACLs in the file system.

Applying default ACLs

Perform the following steps to apply default ACLs on NFS and SMB exports:

1. Create a fileset or directory in the file system as shown in the following example:

```
mkdir -p /ibm/gpfs0/testsmlexport
```
2. Change the owner and group of the fileset or directory using **chown** and **chgrp** respectively. For example:

```
chown -R "DOMAIN\username":"DOMAIN\groupname" /ibm/gpfs0/testsmlexport
```
3. Use the **mmputacl** or **mmeditACL** commands to set the wanted ACE along with specific ACE for owner user and owner group and inheritance flags for the fileset or directory.
4. Check the ACL setting for the fileset or directory by using the **mmgetacl** command.
5. Create the desired NFS or SMB export by using the **mmnfs** or **mmsmb** commands over the fileset or directory.
6. For data exported for SMB clients, it is recommend to manage the ACLs from a Windows clients, since there is already a GUI interface available and the ACL is set according to the requirements of Windows clients. Modifying the ACL directly with **mmputacl** and **mmeditACL** are not advised.

Viewing the owner of the SMB export

Perform the following steps to create an SMB export and view the owner of the export:

1. Submit the **mmsmb export add** command to create SMB export as shown in the following example:

```
mmsmb export add testsmlexport /ibm/gpfs0/testsmlexport
```
2. Issue either **ls -l** command or **mmgetacl** command to view the owner of the export. For example:

```
ls -l /ibm/gpfs0/testsmlexport
```

Or

```
mmgetacl /ibm/gpfs0/testsmlexport
```

Apart from the tasks that are listed earlier in this section, the following table provides a quick overview of the tasks that can be performed to manage ACLs and the corresponding IBM Spectrum Scale command.

Table 20. Commands and reference to manage ACL tasks.

Tasks that can be performed to manage ACLs	Command	Reference topic
Applying ACL at file system, fileset, and export level	mmeditacl	"Applying an existing NFS V4 access control list" on page 194
Inserting ACEs in existing ACLs	mmeditacl	"Changing NFS V4 access control lists" on page 194
Modifying ACLs	mmeditacl	"Changing NFS V4 access control lists" on page 194
Copying Access control list entries	mmeditacl	"Changing NFS V4 access control lists" on page 194
Replacing a complete ACL	mmputacl or mmeditacl	"Changing NFS V4 access control lists" on page 194
Replacing all entries for a specific user inside an ACL	mmeditacl	"Changing NFS V4 access control lists" on page 194
Controlling inheritance of entries inside an ACL	mmputacl or mmeditacl	
Deleting complete ACL	mmdeiacl	"Deleting NFS V4 access control lists" on page 195
Deleting specific ACL entries	mmeditacl	"Changing NFS V4 access control lists" on page 194
Deleting ACL entry for a user	mmeditacl	"Changing NFS V4 access control lists" on page 194
Displaying an ACL	mmgetacl	"Displaying NFS V4 access control lists" on page 194
Changing file system directory's owner and group	chown or chgroup	
Displaying file system directory's owner and group	ls -l or mmgetacl	

Important: The **mmgetacl**, **mmputacl**, and **mmeditacl** commands are available to change the ACLs directly. As the SMB clients might depend on the order of entries in the ACL, it is not recommended to change the ACLs directly on GPFS while using the SMB protocol. Changing an ACL directly in GPFS also does not account for inherited entries. So, it is recommended to change the ACLs from a windows client.

Managing ACLs from Windows clients

For SMB exports, it is recommended to manage the ACLs from a Windows client. The following operations are included in creating an SMB export:

1. Create the folder to export in the file system with the **mkdir** command.
2. Change the owner of the exported folder to a user who configures the initial ACLs.
3. Create the export using the **mmsmb export create** command.
4. Using a Windows client machine, access the newly created share as the user specified in step 2.
5. Right-click on the shared folder, and select **Properties**.
6. Select the **Security** tab and then select **Advanced** to navigate to the more detailed view of permissions.
7. Add and remove permissions as required.

Authorizing object users

The Object Storage service of the IBM Spectrum Scale system uses Keystone service for Identity Management. The Identity Management service consists of user authentication and authorization processes.

Access for the object users to the Object Storage projects are controlled by the user roles and container ACLs. Based on the roles defined for the user, object users can be administrative users and non-administrative users. Non-admin users can only perform operations per container based on the container's X-Container-Read and X-Container-Write ACLs. Container ACLs can be defined to limit access to objects in swift containers. Read access can be limited to only allow download, or allow download and listing. Write access allows the user to upload new objects to a container.

You can use an external AD or LDAP server or a local database as the back-end to store and manage user credentials for user authentication. The authorization details such as relation of users with projects and roles are maintained locally by the keystone server. The customer can select the authentication server to be used. For example, if AD is existing in an enterprise deployment and the users in AD are required to access object data, the customer can decide to use AD as the back-end authentication server.

When the back-end authentication server is AD or LDAP, the user management operations such as creating user, deleting user, and so on are the responsibility of the AD or LDAP administrator, who can optionally also be the Keystone server administrator. When local authentication is used for object access, the user management operations are done by the Keystone administrator. In case of authorization, the management tasks such as creating roles, projects, and associating the user with them is done by the Keystone Administrator. The Keystone administration can be done through the Keystone V3 REST API or by using an OpenStack python-based client.

Before you start creating object users, and projects, ensure that Keystone server is configured and the authentication servers are set up properly. You can use the `mmces service list -a -v` command to see whether Keystone is configured properly.

The object users are authorized to the object data and resources by creating and managing roles and ACLs. The roles and ACLs define the actions that can be performed by the user on the object resources such as accessing data, managing the projects, creating projects, read, write, run permissions, and so on.

Configuring container ACLs to authorize object data users

The following examples and sections provide an understanding on how to set up container ACLs and define the access permissions for the user.

Creating containers:

The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

It is the responsibility of the Keystone server administrator to create and manage accounts. The account defines a namespace for containers. A container must be unique within the owning account and account must use a unique name within the project. The `admin` account is created by default.

The following is an example of how to create containers.

GUI navigation

To work with this function in the IBM Spectrum Scale GUI, log on to the GUI and select **Object > Containers**.

1. Issue the **swift post container** command to create a container by using the Swift Command Line Client. In the following example, the Keystone administrator creates a `public_readOnly` container in admin account.

```
# swift post public_readOnly --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
```

2. Issue the **swift list** command to list the containers that are available for the account. In the following example, the system lists the containers that are available in the admin project.

```
# swift list --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
public_readOnly
```

3. Issue the **swift stat** command to list the accounts, containers, or objects details. In the following example, the system displays the admin account details.

```
# swift stat -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
StorageURL: http://tully-ces-ip.adcons.spectrum:8080/v1
/AUTH_bea5a0c632e54eaf85e9150a16c443ce
Auth Token: 1f6260c4f8994581a465b8225075c932
Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
Containers: 1
Objects: 0
Bytes: 0
Containers in policy "policy-0": 1
Objects in policy "policy-0": 0
Bytes in policy "policy-0": 0
X-Account-Project-Domain-Id: default
X-Timestamp: 1432766053.43581
X-Trans-Id: tx9b96c4a8622c40b3ac69a-0055677ce7
Content-Type: text/plain; charset=utf-8
Accept-Ranges: bytes
```

In the following example, the system displays the `public_readOnly` container details, on the admin account:

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly
Auth Token: 957d6c37155b44d3a476441bc927835d
Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
Container: public_readOnly
Objects: 0
Bytes: 0
Read ACL:
Write ACL:
Sync To:
Sync Key:
Accept-Ranges: bytes
X-Storage-Policy: Policy-0
X-Timestamp: 1432795292.10297
X-Trans-Id: tx9b05c2135a9c4034b910c-0055677dad
Content-Type: text/plain; charset=utf-8
```

By default, only users who are having a Keystone role specified in the `proxy-server.conf` `operator_roles` option are allowed to create container on an account.

To list `operators_role` on the IBM Spectrum Scale system during installation, issue the **mmobj config list** command as shown in the following example:

```
mmobj config list --ccrfile proxy-server.conf --section filter:keystoneauth --property operator_roles
```

To list `operators_role` in all other cases, issue the **mmobj config list** with the following parameters:


```
mmobj config list --ccrfile proxy-server.conf --section filter:keystone --property operator_roles
```

Keystone administrator can also use the container to control access to the objects by using an access control list (ACL). The following example shows that when a member of the `admin` account tries to display the details of `public_readOnly` account, the process fails because it does not have an operator role or access control defined:

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username member
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
Container HEAD failed: http://tully-ces-ip.adcons.spectrum:8080/v1
/AUTH_bea5a0c632e54eaf85e9150a16c443ce/public_readOnly 403 Forbidden
```

Related tasks:

“Creating read ACLs to authorize object users”

The Keystone administrator can create container ACLs to grant read permissions using `X-Container-Read` headers in curl tool or `-read-acl` flag in the Swift Command Line Client.

“Creating write ACLs to authorize object users” on page 213

The Keystone administrator can create container ACLs to grant write permissions using `X-Container-Write` headers in the curl tool or `-write-acl` flag in the Swift Command Line Client.

Creating read ACLs to authorize object users:

The Keystone administrator can create container ACLs to grant read permissions using `X-Container-Read` headers in curl tool or `-read-acl` flag in the Swift Command Line Client.

The following example shows how to create read permission in an ACL.

1. Upload the object `imageA.JPG` to `public_readOnly` container as the Keystone administrator.

```
# swift upload public_readOnly imageA.JPG --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
imageA.JPG
```

2. Issue the `swift post` command to provide public read access to the `public_readOnly` container.

```
# swift post public_readOnly --read-acl ".r:*,.rlistings" --os-auth-url http://tully-ces-ip.
adcons.spectrum:35357/v3 --os-project-name admin --os-project-domain-name Default
--os-username admin --os-user-domain-name Default --os-password Passw0rd --auth-version 3
```

Note: The `.r:*` ACL specifies access for any referrer regardless of account affiliation or user name. The `.rlistings` ACL allows to list the containers and read (download) objects.

3. Issue the `swift stat` command at the container level to see the access details.

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly
  Auth Token: 91a27a5ed8dc40d582e71844ca019c32
  Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
  Container: public_readOnly
  Objects: 3
  Bytes: 8167789
  Read ACL: .r:*,.rlistings
  Write ACL:
  Sync To:
  Sync Key:
  Accept-Ranges: bytes
  X-Trans-Id: tx73b0696705b94bf885bd5-0055678ab1
X-Storage-Policy: Policy-0
  X-Timestamp: 1432795292.10297
  Content-Type: text/plain; charset=utf-8
```

4. As the `student` user from the `students` account, list and download the details of `public_readOnly` container that is created in the `admin` account.

Listing the details:

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly
Auth Token: d6ee0fb5e33748b1b9035a3b690c7587
Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
Container: public_readOnly
Objects: 3
Bytes: 8167789
Read ACL:
Write ACL:
Sync To:
Sync Key:
Accept-Ranges: bytes
X-Storage-Policy: Policy-0
X-Timestamp: 1432795292.10297
X-Trans-Id: tx09893920a6154faab6ace-0055678f6d
Content-Type: text/plain; charset=utf-8
```

Listing the container objects:

```
# swift list public_readOnly --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
file.txt
imageA.JPG
imageB.JPG
```

Downloading container objects:

```
# swift download public_readOnly --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
imageB.JPG [auth 0.321s, headers 0.380s, total 0.390s, 37.742 MB/s]
file.txt [auth 0.533s, headers 0.594s, total 0.594s, 0.000 MB/s]
imageA.JPG [auth 0.119s, headers 0.179s, total 18.135s, 0.308 MB/s]
```

- As the *student1* user from the *students* account, receive deny write access, while trying to upload new content in the *public_readOnly* container:

```
# swift upload public_readOnly photo.jpg --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
Warning: failed to create container 'public_readOnly': 403 Forbidden:

Forbidden

Access was denied to this resourc
Object PUT failed: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly/photo.jpg 403 Forbidden
```

Manipulating the read ACLs

The following table list different read ACLs combinations:

Table 21. ACL options that are available to manipulate object read ACLs

Permission	Read ACL options
Read for all referrers	.r:*
Read and list for all referrers and listing	.r:*,.rlistings
Read and list for a user in a specific project	<project_name project_id>:<user_name user_id>

Table 21. ACL options that are available to manipulate object read ACLs (continued)

Permission	Read ACL options
Read and list for a user in every project	*:<user_name user_id>
Read and list for every user in a project	<project_name project_id>:<*>
Read and list for every user in every project	<*>:<*>

Note: Use comma (,) to separate ACLs. For example, `--read-acl admin:admin,students:student1`.

Related tasks:

“Creating containers” on page 209

The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

“Creating write ACLs to authorize object users”

The Keystone administrator can create container ACLs to grant write permissions using X-Container-Write headers in the curl tool or `--write-acl` flag in the Swift Command Line Client.

Creating write ACLs to authorize object users:

The Keystone administrator can create container ACLs to grant write permissions using X-Container-Write headers in the curl tool or `--write-acl` flag in the Swift Command Line Client.

Provides an example on how to configure write ACLs by using curl tool.

1. Create token and proceed to create a container named `writeOnly` with write permissions for `member` user who is part of the `admin` project and `student1` user who is part of the `students` project.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X PUT -H "Content-Length: 0" -H "X-Auth-Token: ${token}" -H
"X-Container-Write: admin:member,students:student1" -H "X-Container-Read: "
HTTP/1.1 201 Created
Content-Length: 0
Content-Type: text/html; charset=UTF-8
X-Trans-Id: txf7b0bfef877345949c61c-005567b9d1
Date: Fri, 29 May 2015 00:58:57 GMT
```

2. Issue a token as `student1` from the `students` project and upload an object by using the curl tool.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly/imageA.JPG -X PUT -H "X-Auth-Token: ${token}" --upload-file imageA.JPG
HTTP/1.1 100 Continue
HTTP/1.1 201 Created
Last-Modified: Fri, 29 May 2015 01:11:28 GMT
Content-Length: 0
Etag: 95d8c44b757f5b0c111750694dffef2b
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx6caa0570bfcd419782274-005567bcbe
Date: Fri, 29 May 2015 01:11:28 GMT
```

3. List the state of the `writeOnly` container as `student1` user of the `students` project. This operation fails as the user does not have the required privileges.

```
# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly/imageA.JPG -X HEAD -H "X-Auth-Token: ${token}"
HTTP/1.1 403 Forbidden
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx4f7dfbfd74204785b6b50-005567bd8c
Content-Length: 0
Date: Fri, 29 May 2015 01:14:52 GMT
```

4. Grant read permissions to *student1* user of the *students* project:

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')
```

```
# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_
bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X POST -H "Content-Length: 0" -H "X-Auth-Token:
${token}" -H "X-Container-Read: students:student1"
HTTP/1.1 204 No Content
Content-Length: 0
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx77aafe0184da4b68a7756-005567beac
Date: Fri, 29 May 2015 01:19:40 GMT
```

5. Verify whether the *student1* user has the read access now.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')
```

```
# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X GET -H "X-Auth-Token: ${token}"
HTTP/1.1 200 OK
Content-Length: 11
X-Container-Object-Count: 1
Accept-Ranges: bytes
X-Storage-Policy: Policy-0
X-Container-Bytes-Used: 5552466
X-Timestamp: 1432861137.91693
Content-Type: text/plain; charset=utf-8
X-Trans-Id: tx246b39018a5c4bcb90c7f-005567bff3
Date: Fri, 29 May 2015 01:25:07 GMT
```

imageA.JPG

Note: Object Storage does not support public write ACLs.

Related tasks:

“Creating containers” on page 209

The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

“Creating read ACLs to authorize object users” on page 211

The Keystone administrator can create container ACLs to grant read permissions using X-Container-Read headers in curl tool or `-read-acl` flag in the Swift Command Line Client.

Authorization limitations

Authorization limitations are specific to the protocols that are used to access data.

NFS ACL limitations

ACLs are stored as NFSv4 ACLs in the file system.

For more information on the known limitations of the NFSV4 ACLs, see “GPFS exceptions and limitations to NFS V4 ACLs” on page 882.

SMB ACL limitations

The following are the SMB ACL limitations:

- ACL of a new child file or directory depends on the ACL type, the file system settings, and the ACL of the parent directory. Depending on these variables, the results in the IBM Spectrum Scale might be slightly different than in Microsoft Windows. For example, if the parent directory is set to have two ACEs, for example full access for owner and for everyone, the Windows default is to create two ACLs for the child; one is to allow full access for owner and other to allow full access for everyone. The IBM Spectrum Scale system by default creates six ACLs to allow and deny ACLs for owner, group, and everyone.
- The special permissions such as Write Data/Create File and Create Folder/Append Data cannot be set separately for files. Enabling one always enables the other, and disabling one always disables the other. If these access control entries (ACEs) are not inherited by files, you can set them separately for directories. You can configure two separate ACEs, where the ACE that is inherited by files has either both special permissions that are enabled or both disabled, and another ACE that is inherited by directories where one of the special permissions is enabled and the other disabled. In this case, the Apply onto field of the Permission Entry panel can contain the following values:
 - This folder only
 - This folder and subfolders
 - Subfolders only
- If domain server manages the UID and GID mapping, the UID and GID mappings must be configured in the domain server before an ACE for that user or group can be created.
- Users and groups that belonged to another domain, and was migrated to a new domain by using the SID-History mechanism, cannot be stored in an ACL.
- Most well-known SIDs and built-in SIDs cannot be stored in an ACL. Only the "Everyone" SID can be stored and used in an IBM Spectrum Scale system.
- The SMB ACLs cannot be modified when LDAP-based authentication is used for file access.
- Microsoft Windows enables you to limit the scope of inheritance for an ACE to one inheritance by selecting the **Apply these permissions to objects and/or containers within this container only** check box in the Windows Explorer. The IBM Spectrum Scale system does not support to configure this option and limit the scope of inheritance for an ACL.
- ACL inheritance stops at fileset junction points; new filesets always have the default ACL (770 root root).
- The root path of every SMB export needs read permission (read data, read attribute, read extended attribute) for everyone, to prevent the unexpected behavior of, for example, Windows Explorer.
- To prevent display of Access Denied errors, the user must have the read attribute permission on all parent directories, when they have access to a file or directory.
- The value of the `dacl_protected` bit related to the Include Inheritable permissions from this object's parent check box can be changed only through SMB. The ACL commands cannot access this field. Setting a new ACL resets this field.
- The commands that are used to work on the ACLs do not support recursive updates of inherited ACEs in the file tree.
- Access privileges defined in Windows are not honored. Those privileges are tied to administrator groups and allow access, where the ACL alone does not grant it.
- Audit and alarm ACEs are not supported inside an ACL.
- The Bypass Traverse Check is implemented in GPFS for SMB clients only. Clients that use other protocols might still be locked out because the parent tree of an export has more restrictive ACLs than the export itself.

- POSIX-style ACLs are not supported.
- Similar to the POSIX standard, to read the content of a subdirectory, apart from the read permission in the ACL of this subdirectory, the user also need to have traversal permission (SEARCH in Windows, EXECUTE in POSIX) for all of the parent directories. You can set the traverse permission in the “Everyone” group ACE at the share root, and inherit this privilege to all subdirectories. For the SMB protocol, this is applicable only if the configuration option *bypassTraversalCheck* is disabled.
- Even though the underlying file system does not enforce the permissions for extended attributes (READ_NAMED and WRITE_NAMED), these are enforced for SMB clients.

ACL limitations that are applicable to all protocols

The following limitations are applicable to all protocols:

- When creating a file system, the user needs to specify `-k nfs4` to specifically use NFSv4 ACLs, otherwise the default `-k all` uses both POSIX ACLs and NFSV4 ACLs.
- The IBM Spectrum Scale Object Storage does not do file share with NFS and SMB.

Chapter 11. GPFS commands

A list of all the GPFS commands and a short description of each is presented in this topic.

Table 22 summarizes the GPFS-specific commands.

Table 22. GPFS commands

Command	Purpose
"gpfs.snap command" on page 221	Creates an informational system snapshot at a single point in time. This system snapshot consists of information such as cluster configuration, disk configuration, network configuration, network status, GPFS logs, dumps, and traces.
"mmaddcallback command" on page 226	Registers a user-defined command that GPFS will execute when certain events occur.
"mmadddisk command" on page 239	Adds disks to a GPFS file system.
"mmaddnode command" on page 245	Adds nodes to a GPFS cluster.
"mmafmconfig command" on page 248	Can be used to manage home caching behavior and mapping of gateways and home NFS exported servers.
"mmafmctl command" on page 251	This command is for various operations and reporting information on all filesets. It is recommended to read the <i>IBM Spectrum Scale: Advanced Administration Guide</i> AFM and AFM Disaster Recovery chapters in conjunction with this manual for detailed description of the functions.
"mmafmlocal command" on page 264	Provides a list of cached files and file statistics such as inode number, allocated blocks, and so on.
"mmapplypolicy command" on page 266	Deletes files, migrates files between storage pools, or does file compression or decompression in a file system as directed by policy rules.
"mmauth command" on page 276	Manages secure access to GPFS file systems.
"mmbackup command" on page 281	Performs a backup of a GPFS file system or independent fileset to a Tivoli Storage Manager (TSM) server.
"mmbackupconfig command" on page 290	Collects GPFS file system configuration information.
"mmbuildgpl command" on page 292	Manages prerequisite packages for Linux and builds the GPFS portability layer.
"mmcallhome command" on page 293	Manages the call home operations.
"mmces command" on page 304	Manages CES configuration.
"mmcesdr command" on page 313	Manages protocol cluster disaster recovery.
"mmchattr command" on page 321	Changes attributes of one or more GPFS files.
"mmchcluster command" on page 327	Changes GPFS cluster configuration data.
"mmchconfig command" on page 331	Changes GPFS configuration parameters.
"mmchdisk command" on page 354	Changes state or parameters of one or more disks in a GPFS file system.
"mmcheckquota command" on page 361	Checks file system user, group and fileset quotas.
"mmchfileset command" on page 365	Changes the attributes of a GPFS fileset.
"mmchfs command" on page 371	Changes the attributes of a GPFS file system.

Table 22. GPFS commands (continued)

Command	Purpose
"mmchlicense command" on page 377	Controls the type of GPFS license associated with the nodes in the cluster.
"mmchmgr command" on page 379	Assigns a new file system manager node or cluster manager node.
"mmchnode command" on page 381	Changes node attributes.
"mmchnodeclass command" on page 385	Changes user-defined node classes.
"mmchnsd command" on page 388	Changes Network Shared Disk (NSD) configuration attributes.
"mmchpolicy command" on page 391	Establishes policy rules for a GPFS file system.
"mmchpool command" on page 394	Modifies storage pool properties.
"mmchqos command" on page 396	Changes the Quality of Service for I/O operations (QoS) settings for a file system.
"mmclone command" on page 400	Creates and manages file clones.
"mmcrcluster command" on page 403	Creates a GPFS cluster from a set of nodes.
"mmcrfileset command" on page 408	Creates a GPFS fileset.
"mmcrfs command" on page 414	Creates a GPFS file system.
"mmcrnodeclass command" on page 424	Creates user-defined node classes.
"mmcrnsd command" on page 426	Creates Network Shared Disks (NSDs) used by GPFS.
"mmcrsnapshot command" on page 431	Creates a snapshot of a file system or fileset at a single point in time.
"mmdefedquota command" on page 434	Sets default quota limits.
"mmdefquotaoff command" on page 437	Deactivates default quota limit usage.
"mmdefquotaon command" on page 440	Activates default quota limit usage.
"mmdefragfs command" on page 443	Reduces disk fragmentation by increasing the number of full free blocks available to the file system.
"mmdelacl command" on page 446	Deletes a GPFS access control list.
"mmdelcallback command" on page 448	Deletes one or more user-defined callbacks from the GPFS system.
"mmdeldisk command" on page 449	Deletes disks from a GPFS file system.
"mmdelfileset command" on page 455	Deletes a GPFS fileset.
"mmdelfs command" on page 458	Removes a GPFS file system.
"mmdelnnode command" on page 460	Removes one or more nodes from a GPFS cluster.
"mmdelnnodeclass command" on page 463	Deletes user-defined node classes.
"mmdelnnsd command" on page 465	Deletes Network Shared Disks (NSDs) from the GPFS cluster.
"mmdelsnapshot command" on page 467	Deletes a GPFS snapshot.
"mmdf command" on page 470	Queries available file space on a GPFS file system.
"mmdiag command" on page 473	Displays diagnostic information about the internal GPFS state on the current node.
"mmeditacl command" on page 478	Creates or changes a GPFS access control list.
"mmedquota command" on page 481	Sets quota limits.
"mmexportfs command" on page 485	Retrieves the information needed to move a file system to a different cluster.
"mmfsck command" on page 487	Checks and repairs a GPFS file system.

Table 22. GPFS commands (continued)

Command	Purpose
"mmfsctl command" on page 496	Issues a file system control request.
"mmgetacl command" on page 500	Displays the GPFS access control list of a file or directory.
"mmgetstate command" on page 503	Displays the state of the GPFS daemon on one or more nodes.
"mmhadoopctl command" on page 506	Installs and sets up the GPFS connector for a Hadoop distribution; starts or stops the GPFS connector daemon on a node.
"mmimgbackup command" on page 508	Performs a backup of a single GPFS file system metadata image.
"mmimgrestore command" on page 511	Restores a single GPFS file system from a metadata image.
"mmimportfs command" on page 513	Imports into the cluster one or more file systems that were created in another GPFS cluster.
"mmlinkfileset command" on page 517	Creates a junction that references the root directory of a GPFS fileset.
"mmlsattr command" on page 519	Queries file attributes.
"mmlscallback command" on page 522	Lists callbacks that are currently registered in the GPFS system.
"mmlscluster command" on page 524	Displays the current configuration information for a GPFS cluster.
"mmlsconfig command" on page 526	Displays the current configuration data for a GPFS cluster.
"mmlsdisk command" on page 528	Displays the current configuration and state of the disks in a file system.
"mmlsfileset command" on page 532	Displays attributes and status for GPFS filesets.
"mmlsfs command" on page 536	Displays file system attributes.
"mmlslicense command" on page 540	Displays information about the GPFS node licensing designation.
"mmlsmgr command" on page 542	Displays which node is the file system manager for the specified file systems or which node is the cluster manager.
"mmlsmount command" on page 544	Lists the nodes that have a given GPFS file system mounted.
"mmlsnodeclass command" on page 546	Displays node classes defined in the system.
"mmlsnsd command" on page 549	Displays Network Shared Disk (NSD) information for the GPFS cluster.
"mmlspolicy command" on page 552	Displays policy information.
"mmlspool command" on page 554	Displays information about the known storage pools.
"mmlsquota command" on page 559	Displays quota information for a user, group, or fileset.
"mmlsqos command" on page 556	Displays the I/O performance values of a file system, when you enable Quality of Service for I/O operations (QoS) with the mmchqos command.
"mmlssnapshot command" on page 563	Displays GPFS snapshot information.
"mmmigratefs command" on page 566	Performs needed conversions to support new file system features.
"mmm mount command" on page 568	Mounts GPFS file systems on one or more nodes in the cluster.
"mmnfs command" on page 570	Manages NFS exports and configuration.
"mmnsdiscover command" on page 579	Rediscovered paths to the specified network shared disks.

Table 22. GPFS commands (continued)

Command	Purpose
"mmobj command" on page 581	Manages configuration of Object protocol service, and administers storage policies for object storage, unified file and object access, and multi-region object deployment.
"mmperfmom command" on page 592	Configures the Performance Monitoring tool and lists the performance metrics.
"mmpmon command" on page 602	Manages performance monitoring and displays performance information.
"mmprotocoltrace command" on page 607	Starts, stops, and monitors tracing for the CES protocols.
"mmpsnap command" on page 611	Creates or deletes identical snapshots on the cache and home clusters, or shows the status of snapshots that have been queued up on the gateway nodes.
"mmputacl command" on page 614	Sets the GPFS access control list for the specified file or directory.
"mmquotaoff command" on page 617	Deactivates quota limit checking.
"mmquotaon command" on page 619	Activates quota limit checking.
"mmremotecluster command" on page 621	Manages information about remote GPFS clusters.
"mmremotefs command" on page 624	Manages information needed for mounting remote GPFS file systems.
"mmrepquota command" on page 627	Displays file system user, group, and filesset quotas.
"mmrestoreconfig command" on page 631	Restores file system configuration information.
"mmrestorefs command" on page 635	Restores a file system or an independent filesset from a snapshot.
"mmrestripefile command" on page 639	Rebalances or restores the replication factor of the specified files, or performs any incomplete or deferred file compression or decompression.
"mmrestripefs command" on page 642	Rebalances or restores the replication factor of all the files in a file system. Alternatively, this command performs any incomplete or deferred file compression or decompression of all the files in a file system.
"mmrpldisk command" on page 648	Replaces the specified disk.
"mmsdrrestore command" on page 655	Restores the latest GPFS system files on the specified nodes.
"mmsetquota command" on page 657	Sets quota limits.
"mmshutdown command" on page 661	Unmounts all GPFS file systems and stops GPFS on one or more nodes.
"mmsmb command" on page 663	Administers SMB exports, export ACLs, and global configuration.
"mmsnapdir command" on page 674	Controls how the special directories that connect to snapshots appear.
"mmstartup command" on page 678	Starts the GPFS subsystem on one or more nodes.
"mmtracectl command" on page 680	Sets up and enables GPFS tracing.
"mmumount command" on page 684	Unmounts GPFS file systems on one or more nodes in the cluster.
"mmunlinkfilesset command" on page 687	Removes the junction to a GPFS filesset.

Table 22. GPFS commands (continued)

Command	Purpose
“mmuserauth command” on page 690	Manages the authentication of protocol users who need to access the protocol data that is stored on the system. You can create, list, verify, and remove authentication configuration using this command.
“mmwinservctl command” on page 709	Manages the mmwinserv Windows service.
“spectrumscale command” on page 711	Installs and configures GPFS; adds nodes to a cluster; deploys and configures protocols, performance monitoring tools, and authentication services; and upgrades GPFS and protocols.

The following commands are specific to GPFS Native RAID and are documented in *IBM Spectrum Scale RAID: Administration*:

- **mmaddcomp**
- **mmaddcompspec**
- **mmaddpdisk**
- **mmchcarrier**
- **mmchcomp**
- **mmchcomploc**
- **mmchenclosure**
- **mmchfirmware**
- **mmchpdisk**
- **mmchrecoverygroup**
- **mmcrrecoverygroup**
- **mmcrvdisk**
- **mmdelcomp**
- **mmdelcomploc**
- **mmdelcompspec**
- **mmdelpdisk**
- **mmdelrecoverygroup**
- **mmdelvdisk**
- **mmdiscovercomp**
- **mmgetdisktopology**
- **mmlscomp**
- **mmlscomploc**
- **mmlscompspec**
- **mmlsenclosure**
- **mmlsfirmware**
- **mmlspdisk**
- **mmlsrecoverygroup**
- **mmlsrecoverygroupevents**
- **mmlsvdisk**
- **mmsyncdisplayid**

gpfs.snap command

Creates an informational system snapshot at a single point in time. This system snapshot consists of information such as cluster configuration, disk configuration, network configuration, network status, GPFS logs, dumps, and traces.

Synopsis

```
gpfs.snap [-d OutputDirectory] [-m | -z]
          [-a | -N {Node[,Node...] | NodeFile | NodeClass}]
          [--check-space | --no-check-space | --check-space-only]
```

gpfs.snap

```
[--cloud-gateway {BASIC | FULL}] [--full-collection] [--deadlock [--quick] |  
--limit-large-files {YYYY:MM:DD:HH:MM | NumberOfDaysBack | latest}]  
[--exclude-aix-disk-attr] [--exclude-aix-lvm] [--exclude-merge-logs]  
[--exclude-net] [--gather-logs] [--mmdf] [--performance] [--prefix]  
[--protocol ProtocolType[,ProtocolType,...]] [--timeout Seconds]  
[--purge-files KeepNumberOfDaysBack]
```

Availability

Available with IBM Spectrum Scale Express Edition or higher.

Description

Use the **gpfs.snap** command as the main tools to gather data when a GPFS problem is encountered, such as a hung file system, a hung GPFS command, or a daemon assert.

The **gpfs.snap** command gathers information (for example, GPFS internal dumps, traces, and kernel thread dumps) to solve a GPFS problem.

Note: By default, large debug files are now a delta collection, which means that they are only collected when there are new files since the previous run of **gpfs.snap**. To override this default behavior, use either the **--limit-large-files** or **--full-collection** options.

Note: This is a service tool and options might change dynamically. The tool impacts performance and occupies disk space when it runs. See the **gpfs.snap command** topic in the *IBM Spectrum Scale: Problem Determination Guide*.

Parameters

-d *OutputDirectory*

Specifies the output directory. The default is `/tmp/gpfs.snapOut`.

-m Specifying this option is equivalent to specifying `--exclude-merge-logs` with `-N`.

-z

Collects **gpfs.snap** data only from the node on which the command is invoked. No master data is collected.

-a

Directs **gpfs.snap** to collect data from all nodes in the cluster. This is the default.

-N {*Node* [, *Node* ...] | *NodeFile* | *NodeClass*}

Specifies the nodes from which to collect **gpfs.snap** data. This option supports all defined node classes. For general information on how to specify node names, see the *Specifying nodes as input to GPFS commands* topic in the *IBM Spectrum Scale: Administration and Programming Reference*.

--check-space

Specifies that space checking is performed before collecting data.

--no-check-space

Specifies that no space checking is performed. This is the default.

--check-space-only

Specifies that only space checking is performed. No data is collected.

--cloud-gateway {**BASIC** | **FULL**}

With the **BASIC** option, the transparent cloud tiering service collects information such as logs, traces, Java™ cores, along with minimal system and IBM Spectrum Scale™ cluster information. No customer sensitive information is collected.

With the **FULL** option, extra details such as Java Heap dump are collected, along with the information captured with the **BASIC** option.

--full-collection

Specifies that all large debug files are collected instead of the default behavior that only collects new files since the previous run of **gpfs.snap**.

--deadlock

Collects only the minimum amount of data necessary to debug a deadlock problem. Part of the data collected is the output of the **mmfsadm dump all** command. This option ignores all other options except for **-a**, **-N**, **-d**, and **--prefix**.

--quick

Collects less data when specified along with the **--deadlock** option. The output includes **mmfsadm dump most**, **mmfsadm dump kthreads**, and 10 seconds of trace in addition to the usual **gpfs.snap** output.

--limit-large-files {YYYY:MM:DD:HH:MM | *NumberOfDaysBack* | **latest**}]

Specifies a time limit to reduce the number of large files collected.

--exclude-aix-disk-attr

Specifies that data about AIX disk attributes will not be collected. Collecting data about AIX disk attributes on an AIX node that has a large number of disks could be very time-consuming, so using this option could help improve performance.

--exclude-aix-lvm

Specifies that data about the AIX Logical Volume Manager (LVM) will not be collected.

--exclude-merge-logs

Specifies that merge logs and waiters will not be collected.

--exclude-net

Specifies that network-related information will not be collected.

--gather-logs

Gathers, merges, and chronologically sorts all of the **mmfs.log** files. The results are stored in the directory specified with **-d** option.

--mmdf

Specifies that **mmdf** output will be collected.

--performance

Specifies that performance data is to be gathered.

Note: The performance script can take up to 30 minutes to run; therefore, it is not included when all other types of protocol information are gathered by default. Specifying this option is the only way to turn on the gathering of performance data.

--prefix

Specifies that the prefix name **gpfs.snap** will be added to the tar file.

--protocol *ProtocolType* [,*ProtocolType*, ...]

Specifies the type (or types) of protocol information to be gathered. By default, whenever any protocol is enabled on a file system, information is gathered for all types of protocol information (except for performance data; see the **--performance** option). However, when the **--protocol** option is specified, the automatic gathering of all protocol information is turned off, and only the specified type of protocol information will be gathered. The following values for *ProtocolType* are accepted:

smb

nfs

object

authentication

ces

core

gpfs.snap

none

--timeout *Seconds*

Specifies the timeout value, in seconds, for all commands.

--purge-files *KeepNumberOfDaysBack*

Specifies that large debug files will be deleted from the cluster nodes based on the *KeepNumberOfDaysBack* value. If 0 is specified, all of the large debug files will be deleted. If a value greater than 0 is specified, large debug files that are older than the number of days specified will be deleted. For example, if the value 2 is specified, the previous two days of large debug files are retained.

This option is not compatible with many of the **gpfs.snap** options because it only removes files and does not collect any **gpfs.snap** data.

Use the **-z** option to generate a non-master snapshot. This is useful if there are many nodes on which to take a snapshot, and only one master snapshot is needed. For a GPFS problem within a large cluster (hundreds or thousands of nodes), one strategy might call for a single master snapshot (one invocation of **gpfs.snap** with no options), and multiple non-master snapshots (multiple invocations of **gpfs.snap** with the **-z** option).

Use the **-N** option to obtain **gpfs.snap** data from multiple nodes in the cluster. When the **-N** option is used, the **gpfs.snap** command takes non-master snapshots of all the nodes specified with this option and a master snapshot of the node on which it was invoked.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **gpfs.snap** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To collect **gpfs.snap** on all nodes with the default data, issue the command:

```
(09:25:47) c34f2n03:~ # gpfs.snap
gpfs.snap started at Mon Feb  8 09:25:54 EST 2016.
Gathering common data.
Gathering Linux specific data...
Gathering trace reports and internal dumps...
gpfs.snap: Spawning remote gpfs.snap calls. Master is c34f2n03.
This may take a while.

Copying file
/tmp/gpfs.snapOut/18720/gpfs.snap.c13c1apv7_0208092648.out.tar.gz from c13c1apv7.gpfs.net ...
gpfs.snap.c13c1apv7_0208092648.out.tar.gz 100% 592KB 592.2KB/s 00:00
Successfully copied file
/tmp/gpfs.snapOut/18720/gpfs.snap.c13c1apv7_0208092648.out.tar.gz from c13c1apv7.gpfs.net.

Copying file
/tmp/gpfs.snapOut/18720/gpfs.snap.c6f2bc4n8_0208092705.out.tar.gz from c6f2bc4n8.gpfs.net ...
gpfs.snap.c6f2bc4n8_0208092705.out.tar.gz 100% 928KB 927.9KB/s 00:00
Successfully copied file
```

```

/tmp/gpfs.snapOut/18720/gpfs.snap.c6f2bc4n8_0208092705.out.tar.gz from c6f2bc4n8.gpfs.net.
Gathering cluster wide protocol data
Packaging master node data.
Writing * to file
/tmp/gpfs.snapOut/18720/collect/gpfs.snap.c34f2n03_master_0208092554.out.tar.gz
Packaging all data.
Writing . to file /tmp/gpfs.snapOut/18720/all.0208092554.tar
gpfs.snap completed at Mon Feb  8 09:26:45 EST 2016
#####
Send file /tmp/gpfs.snapOut/18720/all.0208092554.tar to IBM Service
Examine previous messages to determine additional required data.
#####

```

After this command customer would send the tar file (highlighted) to IBM service as per the message

- To collect **gpfs.snap** on specific nodes, issue the command:

```

(09:32:38) c34f2n03:~ # gpfs.snap -N c34f2n03,c13c1apv7
gpfs.snap started at Mon Feb  8 09:32:48 EST 2016.
Gathering common data.
Gathering Linux specific data...
Gathering trace reports and internal dumps...
gpfs.snap: Spawning remote gpfs.snap calls. Master is c34f2n03.
This may take a while.

```

```

Copying file
/tmp/gpfs.snapOut/23453/gpfs.snap.c13c1apv7_0208093340.out.tar.gz from c13c1apv7.gpfs.net ...
gpfs.snap.c13c1apv7_0208093340.out.tar.gz      100% 583KB 583.1KB/s  00:00
Successfully copied file
/tmp/gpfs.snapOut/23453/gpfs.snap.c13c1apv7_0208093340.out.tar.gz from c13c1apv7.gpfs.net.
Gathering cluster wide protocol data
Packaging master node data.
Writing * to file /tmp/gpfs.snapOut/23453/collect/gpfs.snap.c34f2n03_master_0208093248.out.tar.gz
Packaging all data.
Writing . to file /tmp/gpfs.snapOut/23453/all.0208093248.tar
gpfs.snap completed at Mon Feb  8 09:33:34 EST 2016
#####
Send file /tmp/gpfs.snapOut/23453/all.0208093248.tar to IBM Service
Examine previous messages to determine additional required data.
#####

```

Location

/usr/lpp/mmfs/bin

mmaddcallback command

Registers a user-defined command that GPFS will execute when certain events occur.

Synopsis

```
mmaddcallback CallbackIdentifier --command CommandPathname  
--event Event [, Event...] [--priority Value]  
[--async | --sync [--timeout Seconds] [--onerror Action]]  
[-N {Node [, Node...] | NodeFile | NodeClass}]  
[--parms ParameterString ...]
```

or

```
mmaddcallback {-S Filename | --spec-file Filename}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmaddcallback** command to register a user-defined command that GPFS executes when certain events occur.

The callback mechanism is intended to provide notifications when node and cluster events occur. Invoking complex or long-running commands, or commands that involve GPFS files, may cause unexpected and undesired results, including loss of file system availability. This is particularly true when the **--sync** option is specified.

Note: For documentation about local events (callbacks) and variables for GPFS Native RAID, see the separate publication *IBM Spectrum Scale RAID: Administration*.

Parameters

CallbackIdentifier

Specifies a user-defined unique name that identifies the callback. It can be up to 255 characters long. It cannot contain special characters (for example, a colon, semicolon, blank, tab, or comma) and it cannot start with the letters `gpfs` or `mm` (which are reserved for GPFS internally defined callbacks).

--command *CommandPathname*

Specifies the full path name of the executable to run when the event occurs. On Windows, *CommandPathname* must be a Korn shell script because it will be invoked in the Cygwin **ksh** environment.

The executable called by the callback facility must be installed on all nodes on which the callback can be triggered. Place the executable in a local file system (not in a GPFS file system) so that it is accessible even when the GPFS file system is unavailable.

--event *Event* [, *Event...*]

Specifies a list of events that trigger the callback. The value defines when the callback is invoked. There are two kinds of events: global events and local events. A global event triggers a callback on all nodes in the cluster, such as a **nodeLeave** event, which informs all nodes in the cluster that a node has failed. A local event triggers a callback only on the node on which the event occurred, such as mounting a file system on one of the nodes.

Table 23 on page 231 lists the supported global events and their parameters.

Table 24 on page 232 lists the supported local events and their parameters.

Local events for GPFS Native RAID are documented in *IBM Spectrum Scale RAID: Administration*.

--priority *Value*

Specifies a floating point number that controls the order in which callbacks for a given event are run. Callbacks with a smaller numerical value are run before callbacks with a larger numerical value. Callbacks that do not have an assigned priority are run last. If two callbacks have the same priority, the order in which they are run is undetermined.

--async | **--sync** [**--timeout** *Seconds*] [**--onerror** *Action*]

Specifies whether GPFS will wait for the user program to complete and for how long it will wait. The default is **--async** (GPFS invokes the command asynchronously). **--onerror** *Action* specifies one of the following actions that GPFS is to take if the callback command returns a nonzero error code:

continue

GPFS ignores the result from executing the user-provided command. This is the default.

quorumLoss

The node executing the user-provided command will voluntarily resign as, or refrain from taking over as, cluster manager. This action is valid only in conjunction with the **tiebreakerCheck** event.

shutdown

GPFS will be shut down on the node executing the user-provided command.

-N {*Node* [, *Node* ...] | *NodeFile* | *NodeClass*}

Defines the set of nodes on which the callback is invoked. For global events, the callback is invoked only on the specified set of nodes. For local events, the callback is invoked only if the node on which the event occurred is one of the nodes specified by the **-N** option. The default is **-N all**.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

--parms *ParameterString* ...

Specifies parameters to be passed to the executable specified with the **--command** parameter. The **--parms** parameter can be specified multiple times.

When the callback is invoked, the combined parameter string is tokenized on white-space boundaries. Constructs of the form *%name* and *%name.qualifier* are assumed to be GPFS variables and are replaced with their appropriate values at the time of the event. If a variable does not have a value in the context of a particular event, the string **UNDEFINED** is returned instead.

GPFS recognizes the following variables:

%blockLimit

Specifies the current hard quota limit in KB.

%blockQuota

Specifies the current soft quota limit in KB.

%blockUsage

Specifies the current usage in KB for quota-related events.

%ccrObjectName

Specifies the name of the modified object.

%ccrObjectValue

Specifies the value of the modified object.

%ccrObjectVersion

Specifies the version of the modified object.

%clusterManager [*.qualifier*]

Specifies the current cluster manager node.

%clusterName

Specifies the name of the cluster where this callback was triggered.

mmaddcallback

	%ckDataLen
	Specifies the length of data involved in a checksum mismatch.
	%ckErrorCountClient
	Specifies the cumulative number of errors for the client side in a checksum mismatch.
	%ckErrorCountNSD
	Specifies the cumulative number of errors for the NSD side in a checksum mismatch.
	%ckErrorCountServer
	Specifies the cumulative number of errors for the server side in a checksum mismatch.
	%ckNSD
	Specifies the NSD involved.
	%ckOtherNode
	Specifies the IP address of the other node in an NSD checksum event.
	%ckReason
	Specifies the reason string indicating why a checksum mismatch callback was invoked.
	%ckReportingInterval
	Specifies the error-reporting interval in effect at the time of a checksum mismatch.
	%ckRole
	Specifies the role (client or server) of a GPFS node.
	%ckStartSector
	Specifies the starting sector of a checksum mismatch.
	%daName
	Specifies the name of the declustered array involved.
	%daRemainingRedundancy
	Specifies the remaining fault tolerance in a declustered array.
	%diskName
	Specifies a disk or a comma-separated list of disk names for which this callback is triggered.
	%downNodes[.qualifier]
	Specifies a comma-separated list of nodes that are currently down. Only nodes local to the given cluster are listed. Nodes which are in a remote cluster but have temporarily joined the cluster are not included.
	%eventName
	Specifies the name of the event that triggered this callback.
	%eventNode[.qualifier]
	Specifies a node or comma-separated list of nodes on which this callback is triggered. Note that the list may include nodes which are not local to the given cluster, but have temporarily joined the cluster to mount a file system provided by the local cluster. Those remote nodes could leave the cluster if there is a node failure or if the file systems are unmounted.
	%filesLimit
	Specifies the current hard quota limit for the number of files.
	%filesQuota
	Specifies the current soft quota limit for the number of files.
	%filesUsage
	Specifies the current number of files for quota-related events.
	%filessetName
	Specifies the name of a fileset for which the callback is being executed.

	%filesetSize	Specifies the size of the fileset.
	%fsErr	Specifies the file system structure error code.
	%fsName	Specifies the file system name for file system events.
	%hardLimit	Specifies the hard limit for the block.
	%homeServer	Specifies the name of the home server.
	%inodeLimit	Specifies the hard limit of the inode.
	%inodeQuota	Specifies the soft limit of the inode.
	%inodeUsage	Specifies the total number of files in the fileset.
	%myNode[.qualifier]	Specifies the node where callback script is invoked.
	%nodeName	Specifies the node name to which the request is sent.
	%nodeNameList	Specifies a space-separated list of node names to which the request is sent.
	%pcacheEvent	Specifies the pcache related events.
	%pdFru	Specifies the FRU (field replaceable unit) number of the pdisk.
	%pdLocation	The physical location code of a pdisk.
	%pdName	The name of the pdisk involved.
	%pdPath	The block device path of the pdisk.
	%pdPriority	The replacement priority of the pdisk.
	%pdState	The state of the pdisk involved.
	%pdWwn	The worldwide name of the pdisk.
	%prepopAlreadyCachedFiles	Specifies the number of files that are cached. These number of files are not read into cache because data is same between cache and home.
	%prepopCompletedReads	Specifies the number of reads executed during a prefetch operation.
	%prepopData	Specifies the total data read from the home as part of a prefetch operation.

mmaddcallback

| **%prepopFailedReads**
| Specifies the number of files for which prefetch failed. Messages are logged to indicate the failure.
| However, there is no indication about the file names that failed to read.

| **%quorumNodes[.qualifier]**
| Specifies a comma-separated list of quorum nodes.

| **%quotaEventType**
| Specifies either the **blockQuotaExceeded** event or the **inodeQuotaExceeded** event. These events are
| related to soft quota limit being exceeded,

| **%quotaID**
| Specifies the numerical ID of the quota owner (UID, GID, or fileset ID).

| **%quotaOwnerName**
| Specifies the name of the quota owner (user name, group name, or fileset name).

| **%quotaType**
| Specifies the type of quota for quota-related events. Possible values are **USR**, **GRP**, or **FILESET**.

| **%reason**
| Specifies the reason for triggering the event. For the **preUnmount** and **unmount** events, the
| possible values are **normal** and **forced**. For the **preShutdown** and **shutdown** events, the possible
| values are **normal** and **abnormal**. For all other events, the value is **UNDEFINED**.

| **%requestType**
| Specifies the type of request to send to the target nodes.

| **%rgCount**
| The number of recovery groups involved.

| **%rgErr**
| A code from a recovery group, where 0 indicates no error.

| **%rgName**
| The name of the recovery group involved.

| **%rgReason**
| The reason string indicating why a recovery group callback was invoked.

| **%senseDataFormatted**
| Sense data for the specific fileset structure error in a formatted string output.

| **%senseDataHex**
| Sense data for the specific fileset structure error in Big endian hex output.

| **%snapshotID**
| Specifies the identifier of the new snapshot.

| **%snapshotName**
| Specifies the name of the new snapshot.

| **%softLimit**
| Specifies the soft limit of the block.

| **%storagePool**
| Specifies the storage pool name for space-related events.

| **%upNodes[.qualifier]**
| Specifies a comma-separated list of nodes that are currently up. Only nodes local to the given
| cluster are listed. Nodes which are in a remote cluster but have temporarily joined the cluster are
| not included.

| **%userName**
| Specifies the user name.

%waiterLength

Specifies the length of the waiter in seconds.

Variables recognized by GPFS Native RAID are documented in *IBM Spectrum Scale RAID: Administration*.

Variables that represent node identifiers accept an optional qualifier that can be used to specify how the nodes are to be identified. When specifying one of these optional qualifiers, separate it from the variable with a period, as shown here:

variable.qualifier

The value for *qualifier* can be one of the following:

ip Specifies that GPFS should use the nodes' IP addresses.

name

Specifies that GPFS should use fully-qualified node names. This is the default.

shortName

Specifies that GPFS should strip the domain part of the node names.

Events and supported parameters

Table 23. Global events and supported parameters

Global event	Supported parameters
afmFilesetExpired Triggered when the contents of a fileset expire either as a result of the fileset being disconnected for the expiration timeout value or when the fileset is marked as expired using the AFM administration commands.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmFilesetUnexpired Triggered when the contents of a fileset become unexpired either as a result of the reconnection to home or when the fileset is marked as unexpired using the AFM administration commands.	%fsName %filesetName %pcacheEvent %homeServer %reason
nodeJoin Triggered when one or more nodes join the cluster.	%eventNode
nodeLeave Triggered when one or more nodes leave the cluster.	%eventNode
quorumReached Triggered when a quorum has been established in the GPFS cluster. This event is triggered only on the cluster manager, not on all the nodes in the cluster.	%quorumNodes
quorumLoss Triggered when quorum has been lost in the GPFS cluster.	N/A
quorumNodeJoin Triggered when one or more quorum nodes join the cluster.	%eventNode
quorumNodeLeave Triggered when one or more quorum nodes leave the cluster.	%eventNode

mmaddcallback

Table 23. Global events and supported parameters (continued)

Global event	Supported parameters
clusterManagerTakeOver Triggered when a new cluster manager node is elected. This happens when a cluster first starts up or when the current cluster manager fails or resigns and a new node takes over as cluster manager.	N/A

Table 24. Local events and supported parameters

Local event	Supported parameters
afmCmdRequeued Triggered during replication when messages are queued up again to be retried later. Queued messages are retried every 15 minutes.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmFilesetUnmounted Triggered when the fileset is moved to an Unmounted state because NFS server is not reachable or remote cluster mount is not available for GPFS Native protocol.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmHomeConnected Triggered when a gateway node connects to the afmTarget of the fileset that it is serving. This event is local on gateway nodes.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmHomeDisconnected Triggered when a gateway node gets disconnected from the afmTarget of the fileset that it is serving. This event is local on gateway nodes.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmManualResyncComplete Triggered when a manual resync is completed.	%fsName %filesetName %reason
afmPrepopEnd Triggered when all the files specified by a prefetch operation have been cached successfully. This event is local on gateway nodes.	%fsName %filesetName %prepopCompletedReads %prepopFailedReads %prepopAlreadyCachedFiles %prepopData
afmQueueDropped Triggered when replication encounters an issue that cannot be corrected. After the queue is dropped, next recovery action attempts to fix the error and continue to replicate.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmRecoveryFail Triggered when recovery fails. The recovery action is retried after 300 seconds. If recovery keeps failing, fileset is moved to a resync state if the fileset mode allows it.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmRecoveryStart Triggered when AFM recovery starts. This event is local on gateway nodes.	%fsName %filesetName %pcacheEvent %homeServer %reason

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
afmRecoveryEnd Triggered when AFM recovery ends. This event is local on gateway nodes.	%fsName %filesetName %pcacheEvent %homeServer %reason
afmRPOMiss Triggered when Recovery Point Objective (RPO) is missed on DR primary filesets, RPO Manager keeps retrying the snapshots. This event occurs when there is lot of data to replicate for the RPO snapshot to be taken or there is an error such as, deadlock and recovery keeps failing.	%fsName %filesetName %pcacheEvent %homeServer %reason
ccrFileChange Triggered when CCR fput operation takes place.	%ccrObjectName %ccrObjectVersion
ccrVarChange Triggered when CCR vput operation takes place.	%ccrObjectName %ccrObjectValue %ccrObjectVersion
daRebuildFailed The daRebuildFailed callback is generated when the spare space in a declustered array has been exhausted, and vdisk tracks involving damaged pdisks can no longer be rebuilt. The occurrence of this event indicates that fault tolerance in the declustered array has become degraded and that disk maintenance should be performed immediately. The daRemainingRedundancy parameter indicates how much fault tolerance remains in the declustered array.	%myNode %rgName %daName %daRemainingRedundancy
deadlockDetected Triggered when a node detects a potential deadlock. If the exit code of the registered callback for this event is 1, debug data will not be collected. See the <code>/usr/lpp/mmfs/samples/deadlockdetected.sample</code> file for an example of using the deadlockDetected event.	%eventName %myNode %waiterLength
deadlockOverload Triggered when an overload event occurs. The event is local to the node detecting the overload condition.	%eventName %nodeName
diskFailure Triggered on the file system manager when the status of a disk in a file system changes to down .	%eventName %diskName %fsName
filesetLimitExceeded Triggered when the file system manager detects that a fileset quota has been exceeded. This is a variation of softQuotaExceeded that applies only to fileset quotas. It exists only for compatibility (and may be deleted in a future version); therefore, using softQuotaExceeded is recommended instead.	%filesetName %fsName %filesetSize %softLimit %hardLimit %inodeUsage %inodeQuota %inodeLimit %quotaEventType

mmaddcallback

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
<p>fsstruct</p> <p>Triggered when the file system manager detects a file system structure (FS Struct) error.</p> <p>For more information about FS Struct errors, see the following topics in the <i>IBM Spectrum Scale: Problem Determination Guide</i>:</p> <ul style="list-style-type: none"> • <i>MMFS_FSSTRUCT</i> • <i>Reliability, Availability, and Serviceability (RAS) events</i> • <i>Information to collect before contacting the IBM Support Center</i> 	<p>%fsName %fsErr %senseDataFormatted %senseDataHex</p>
<p>healthCollapse</p> <p>Triggered when the node health declines below the healthCollapseThreshold long enough for the health check thread to notice.</p>	<p>N/A</p>
<p>lowDiskSpace</p> <p>Triggered when the file system manager detects that disk space usage has reached the high occupancy threshold that is specified in the current policy rule. The event is generated every two minutes until the condition no longer exists. For more information, see the topic <i>Using thresholds with external pools</i> in the <i>IBM Spectrum Scale: Administration and Programming Reference</i>.</p>	<p>%storagePool %fsName</p>
<p>noDiskSpace</p> <p>Triggered when the file system encounters a disk, or storage pool that has run out of space or an inodespace has run out of inodes. An inode space can be an entire file system or an independent fileset. Use the noSpaceEventInterval configuration attribute of the mmchconfig command to control the time interval between two noDiskSpace events. The default value is 120 seconds.</p> <p>When a storage pool runs out of disk space, %reason is “diskspace”, %storagePool is the name of the pool that ran out of disk space, and %filesetName is “UNDEFINED”.</p> <p>When a fileset runs out of inode space, %reason is “inodespace”, %filesetName is the name of the independent fileset that owns the affected inode space, and %storagePool is “UNDEFINED”.</p>	<p>%storagePool %fsName %reason %filesetName</p>

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
<p>nsdCksumMismatch</p> <p>The nsdCksumMismatch callback is generated whenever transmission of vdisk data by the NSD network layer fails to verify the data checksum. This can indicate problems in the network between the GPFS client node and a recovery group server. The first error between a given client and server generates the callback; subsequent callbacks are generated for each ckReportingInterval occurrence.</p>	<p>%myNode %ckRole %ckOtherNode %ckNSD %ckReason %ckStartSector %ckDataLen %ckErrorCountClient %ckErrorCountServer %ckErrorCountNSD %ckReportingInterval</p>
<p>pdFailed</p> <p>The pdFailed callback is generated whenever a pdisk in a recovery group is marked as dead, missing, failed, or readonly.</p>	<p>%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdState</p>
<p>pdPathDown</p> <p>The pdPathDown callback is generated whenever one of the block device paths to a pdisk disappears or becomes inoperative. The occurrence of this event can indicate connectivity problems with the JBOD array in which the pdisk resides.</p>	<p>%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdPath</p>
<p>pdReplacePdisk</p> <p>The pdReplacePdisk callback is generated whenever a pdisk is marked for replacement according to the replace threshold setting of the declustered array in which it resides.</p>	<p>%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdState %pdPriority</p>
<p>pdRecovered</p> <p>The pdRecovered callback is generated whenever a missing pdisk is rediscovered.</p> <p>The following parameters are available to this callback: %myNode, %rgName, %daName, %pdName, %pdLocation, %pdFru, and %pdWwn.</p>	<p>%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn</p>
<p>preMount, preUnmount, mount, unmount</p> <p>These events are triggered when a file system is about to be mounted or unmounted or has been mounted or unmounted successfully. These events are generated for explicit mount and unmount commands, a remount after GPFS recovery and a forced unmount when GPFS panics and shuts down.</p>	<p>%fsName %reason</p>
<p>preRGRelinquish</p> <p>The preRGRelinquish callback is invoked on a recovery group server prior to relinquishing service of recovery groups. The rgName parameter may be passed into the callback as the keyword value _ALL_, indicating that the recovery group server is about to relinquish service for all recovery groups it is serving; the rgCount parameter will be equal to the number of recovery groups being relinquished. Additionally, the callback will be invoked with the rgName of each individual recovery group and an rgCount of 1 whenever the server relinquishes serving recovery group rgName.</p>	<p>%myNode %rgName %rgErr %rgCount %rgReason</p>

mmaddcallback

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
<p>preRGTakeover</p> <p>The preRGTakeover callback is invoked on a recovery group server prior to attempting to open and serve recovery groups. The rgName parameter may be passed into the callback as the keyword value _ALL_, indicating that the recovery group server is about to open multiple recovery groups; this is typically at server startup, and the parameter rgCount will be equal to the number of recovery groups being processed. Additionally, the callback will be invoked with the rgName of each individual recovery group and an rgCount of 1 whenever the server checks to determine whether it should open and serve recovery group rgName.</p>	<p>%myNode %rgName %rgErr %rgCount %rgReason</p>
<p>preShutdown</p> <p>Triggered when GPFS detects a failure and is about to shut down.</p>	<p>%reason</p>
<p>preStartup</p> <p>Triggered after the GPFS daemon completes its internal initialization and joins the cluster, but before the node runs recovery for any file systems that were already mounted, and before the node starts accepting user initiated sessions.</p>	<p>N/A</p>
<p>postRGRelinquish</p> <p>The postRGRelinquish callback is invoked on a recovery group server after it has relinquished serving recovery groups. If multiple recovery groups have been relinquished, the callback will be invoked with rgName keyword _ALL_ and an rgCount equal to the total number of involved recovery groups. The callback will also be triggered for each individual recovery group.</p>	<p>%myNode %rgName %rgErr %rgCount %rgReason</p>
<p>postRGTakeover</p> <p>The postRGTakeover callback is invoked on a recovery group server after it has checked, attempted, or begun to serve a recovery group. If multiple recovery groups have been taken over, the callback will be invoked with rgName keyword _ALL_ and an rgCount equal to the total number of involved recovery groups. The callback will also be triggered for each individual recovery group.</p>	<p>%myNode %rgName %rgErr %rgCount %rgReason</p>
<p>rgOpenFailed</p> <p>The rgOpenFailed callback will be invoked on a recovery group server when it fails to open a recovery group that it is attempting to serve. This may be due to loss of connectivity to some or all of the disks in the recovery group; the rgReason string will indicate why the recovery group could not be opened.</p>	<p>%myNode %rgName %rgErr %rgReason</p>

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
<p>rgPanic</p> <p>The rgPanic callback will be invoked on a recovery group server when it is no longer able to continue serving a recovery group. This may be due to loss of connectivity to some or all of the disks in the recovery group; the rgReason string will indicate why the recovery group can no longer be served.</p>	%myNode %rgName %rgErr %rgReason
<p>sendRequestToNodes</p> <p>Triggered when a node sends a request for collecting expel-related debug data.</p> <p>For this event, the %requestType is requestExpelData.</p>	%eventName %requestType %nodeNames
<p>shutdown</p> <p>Triggered when GPFS completes the shutdown.</p>	%reason
<p>snapshotCreated</p> <p>Triggered after a snapshot is created, and run before the file system is resumed. This event helps correlate the timing of DMAPI events with the creation of a snapshot. GPFS must wait for snapshotCreated to exit before it resumes the file system, so the ordering of DMAPI events and snapshot creation is known.</p> <p>The %filesetName is the name of the fileset whose snapshot was created. For file system level snapshots that affect all filesets, %filesetName is set to global.</p>	%snapshotID %snapshotName %fsName %filesetName
<p>softQuotaExceeded</p> <p>Triggered when the file system manager detects that a soft quota limit (for either files or blocks) has been exceeded. This event is triggered only on the file system manager. Therefore, this event must be handled on all manager nodes.</p>	
<p>startup</p> <p>Triggered after a successful GPFS startup before the node is ready for user initiated sessions. After this event is triggered GPFS proceeds to finish starting including mounting all file systems defined to mount on startup.</p>	N/A
<p>tiebreakerCheck</p> <p>Triggered when the cluster manager detects a lease timeout on a quorum node before GPFS runs the algorithm that decides if the node will remain in the cluster. This event is generated only in configurations that use tiebreaker disks.</p>	N/A
<p>traceConfigChanged</p> <p>Triggered when GPFS tracing configuration is changed.</p>	N/A

mmaddcallback

Table 24. Local events and supported parameters (continued)

Local event	Supported parameters
usageUnderSoftQuota Triggered when the file system manager detects that quota usage has dropped below soft limits and grace time is reset.	%fsName %filesetName %fsName %quotaId %quotaType %quotaOwnerName %blockUsage %blockQuota %blockLimit %filesUsage %filesQuota %filesLimit

Options

-S *Filename* | **--spec-file** *Filename*

Specifies a file with multiple callback definitions, one per line. The first token on each line must be the callback identifier.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmaddcallback** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To register command **/tmp/myScript** to run after GPFS startup, issue this command:

```
mmaddcallback test1 --command=/tmp/myScript --event startup
```

The system displays information similar to:

```
mmaddcallback: Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

2. To register a callback on the NFS servers to export or to unexport a particular file system after it has been mounted or before it has been unmounted, issue this command:

```
mmaddcallback NFSexport --command /usr/local/bin/NFSexport --event mount,preUnmount -N nfserver1,  
nfserver2 --parms "%eventName %fsName" --parms "%eventName %fsName"
```

The system displays information similar to:

```
mmaddcallback: 6027-1371 Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

See also

- “**mmdelcallback** command” on page 448
- “**mmlscallback** command” on page 522

Location

/usr/lpp/mmfs/bin

mmadddisk command

Adds disks to a GPFS file system.

Synopsis

```
mmadddisk Device {"DiskDesc[;DiskDesc...]" | -F StanzaFile} [-a] [-r]
[-v {yes | no}] [-N {Node[,Node...]} | NodeFile | NodeClass]
[--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmadddisk** command to add disks to a GPFS file system. When the **-r** flag is specified, the command rebalances an existing file system after it adds the disks. The command does not require the file system to be mounted. The file system can be in use.

The actual number of disks in your file system might be constrained by products other than GPFS that you installed. See to the individual product documentation.

To add disks to a GPFS file system, first decide which of the following two tasks you want to perform:

1. Create new disks with the **mmcrnsd** command.

In this case, you must also decide whether to create a new set of NSD and pools stanzas or use the rewritten NSD and pool stanzas that the **mmcrnsd** command produces. In a rewritten file, the disk usage, failure group, and storage pool values are the same as the values that are specified in the **mmcrnsd** command.

2. Select disks no longer in use in any file system. To display the disks that are not in use, run the following command:

```
mmfnsd -F
```

command to display the available disks.

Before GPFS 3.5, disk information was specified with disk descriptors. See the following line for the format of a disk descriptor. The second, third, and sixth fields are reserved:

```
DiskName:::DiskUsage:FailureGroup:::StoragePool:
```

For compatibility with earlier versions, the **mmadddisk** command still accepts the traditional disk descriptors, but their use is discouraged.

Note: If **mmadddisk** fails with a **NO_SPACE** error, try one of the following actions:

- Rebalance the file system.
- Run the command **mmfsck -y** to deallocate unreferenced subblocks.
- Create a pool with larger disks and move data from the old pool to the new one.

Parameters

Device

The device name of the file system to which the disks are added. File system names need not be fully qualified. **fs0** is as acceptable as **/dev/fs0**.

This parameter must be first.

mmaddisk

DiskDesc

A descriptor for each disk to be added. Each descriptor is delimited by a semicolon (;) and the entire list must be enclosed in quotation marks (' or "). The use of disk descriptors is discouraged.

-F *StanzaFile*

Specifies a file that contains the NSD stanzas and pool stanzas for the disks to be added to the file system.

NSD stanzas have this format:

```
%nsd:  
nsd=NsdName  
usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}  
failureGroup=FailureGroup  
pool=StoragePool  
servers=ServerList  
device=DiskName
```

where:

nsd=*NsdName*

The name of an NSD previously created by the **mmcrnsd** command. For a list of available disks, run the **mmisnsd -F** command. This clause is mandatory for the **mmaddisk** command.

usage={**dataOnly** | **metadataOnly** | **dataAndMetadata** | **descOnly**}

Specifies the type of data to be stored on the disk:

dataAndMetadata

Indicates that the disk contains both data and metadata. This value is the default for disks in the system pool.

dataOnly

Indicates that the disk contains data and does not contain metadata. This value is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disk contains metadata and does not contain data.

descOnly

Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain disaster recovery configurations. For more information, see the *IBM Spectrum Scale: Advanced Administration Guide* and search for "Synchronous mirroring utilizing GPFS replication"

failureGroup=*FailureGroup*

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

pool=*StoragePool*

Specifies the storage pool to which the disk is to be assigned. If this name is not provided, the default is **system**.

Only the system storage pool can contain **metadataOnly**, **dataAndMetadata**, or **descOnly** disks. Disks in other storage pools must be **dataOnly**.

servers=ServerList

A comma-separated list of NSD server nodes. This clause is ignored by the **mmaddisk** command.

device=DiskName

The block device name of the underlying disk device. This clause is ignored by the **mmaddisk** command.

Note: An NSD belonging to a tiebreaker disk is not allowed to be added to a file system if NSD format conversion is required.

Pool stanzas have this format:

```
%pool:
  pool=StoragePoolName
  blockSize=BlockSize
  usage={dataOnly | metadataOnly | dataAndMetadata}
  layoutMap={scatter | cluster}
  allowWriteAffinity={yes | no}
  writeAffinityDepth={0 | 1 | 2}
  blockGroupFactor=BlockGroupFactor
```

where:

pool=StoragePoolName

Is the name of a storage pool.

blockSize=BlockSize

Specifies the block size of the disks in the storage pool.

usage={dataOnly | metadataOnly | dataAndMetadata}

Specifies the type of data to be stored in the storage pool:

dataAndMetadata

Indicates that the disks in the storage pool contain both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disks contain data and do not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disks contain metadata and do not contain data.

layoutMap={scatter | cluster}

Specifies the block allocation map type. When allocating blocks for a given file, GPFS first uses a round-robin algorithm to spread the data across all disks in the storage pool. After a disk is selected, the location of the data block on the disk is determined by the block allocation map type. If **cluster** is specified, GPFS attempts to allocate blocks in clusters. Blocks that belong to a particular file are kept adjacent to each other within each cluster. If **scatter** is specified, the location of the block is chosen randomly.

The **cluster** allocation method may provide better disk performance for some disk subsystems in relatively small installations. The benefits of clustered block allocation diminish when the number of nodes in the cluster or the number of disks in a file system increases, or when the file system's free space becomes fragmented. The **cluster** allocation method is the default for GPFS clusters with eight or fewer nodes and for file systems with eight or fewer disks.

The **scatter** allocation method provides more consistent file system performance by averaging out performance variations due to block location (for many disk subsystems, the location of the data

mmaddisk

relative to the disk edge has a substantial effect on performance). This allocation method is appropriate in most cases and is the default for GPFS clusters with more than eight nodes or file systems with more than eight disks.

The block allocation map type cannot be changed after the storage pool has been created.

allowWriteAffinity={yes | no}

Indicates whether the GPFS File Placement Optimizer (FPO) feature is to be enabled for the storage pool. See the section about the GPFS File Placement Optimizer in the *IBM Spectrum Scale: Advanced Administration Guide* for additional information.

writeAffinityDepth={0 | 1 | 2}

Specifies the allocation policy to be used by the node writing the data.

A write affinity depth of 0 indicates that each replica is to be striped across the disks in a cyclical fashion with the restriction that no two disks are in the same failure group. By default, the unit of striping is a block; however, if the block group factor is specified in order to exploit chunks, the unit of striping is a chunk.

A write affinity depth of 1 indicates that the first copy is written to the writer node. The second copy is written to a different rack. The third copy is written to the same rack as the second copy, but on a different half (which can be composed of several nodes).

A write affinity depth of 2 indicates that the first copy is written to the writer node. The second copy is written to the same rack as the first copy, but on a different half (which can be composed of several nodes). The target node is determined by a hash value on the fileset ID of the file, or it is chosen randomly if the file does not belong to any fileset. The third copy is striped across the disks in a cyclical fashion with the restriction that no two disks are in the same failure group.

This behavior can be altered on an individual file basis by using the **--write-affinity-failure-group** option of the **mmchattr** command.

This parameter is ignored if write affinity is disabled for the storage pool.

blockGroupFactor=BlockGroupFactor

Specifies how many file system blocks are laid out sequentially on disk to behave like a single large block. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

See the section about File Placement Optimizer in the *IBM Spectrum Scale: Advanced Administration Guide*.

- a Specifies asynchronous processing. If this flag is specified, the **mmaddisk** command returns after the file system descriptor is updated and the rebalancing scan is started; it does not wait for rebalancing to finish. If no rebalancing is requested (the **-r** flag not specified), this option has no effect.
- r Rebalance all existing files in the file system to use the new disks.

Note: Rebalancing of files is an I/O intensive and time-consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance the file system over time.

-v {yes | no}

Verify that specified disks do not belong to an existing file system. The default is **-v yes**. Specify **-v no** only when you want to reuse disks that are no longer needed for an existing file system. If the command is interrupted for any reason, use the **-v no** option on the next invocation of the command.

Important: Using **-v no** on a disk that already belongs to a file system corrupts that file system. This problem is not detected until the next time that file system is mounted.

-N {Node[,Node...]} | NodeFile | NodeClass}

Specifies the nodes that are to participate in the restriping of the file system after the specified disks

are available for use by GPFS. This parameter must be used with the **-r** option. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure occurred.

Security

You must have root authority to run the **mmaddisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. Assume that the file `./newNSDstanza` contains the following NSD stanza:

```
%nsd: nsd=gpfs10nsd
      servers=k148n07,k148n06
      usage=dataOnly
      failureGroup=5
      pool=pool2
```

To add the disk that is defined in this stanza, run the following command:

```
mmaddisk fs1 -F ./newNSDstanza -r
```

The command displays information like the following example:

```
GPFS: 6027-531 The following disks of fs1 will be formatted on node
k148n07.kgn.ibm.com:
  gpfs10nsd: size 2202 MB
Extending Allocation Map
Creating Allocation Map for storage pool 'pool2'
 75 % complete on Thu Feb 16 13:57:52 2006
100 % complete on Thu Feb 16 13:57:54 2006
```

mmaddisk

```
Flushing Allocation Map for storage pool 'pool2'  
GPFS: 6027-535 Disks up to size 24 GB can be added to storage  
pool pool2.  
Checking allocation map for storage pool system  
 62 % complete on Thu Feb 16 13:58:03 2006  
100 % complete on Thu Feb 16 13:58:06 2006  
Checking allocation map for storage pool pool1  
 62 % complete on Thu Feb 16 13:58:11 2006  
100 % complete on Thu Feb 16 13:58:14 2006  
Checking allocation map for storage pool pool2  
 63 % complete on Thu Feb 16 13:58:19 2006  
100 % complete on Thu Feb 16 13:58:22 2006  
GPFS: 6027-1503 Completed adding disks to file system fs1.  
mmaddisk: 6027-1371 Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.  
Restripping fs1 ...  
GPFS: 6027-589 Scanning file system metadata, phase 1 ...  
GPFS: 6027-552 Scan completed successfully.  
GPFS: 6027-589 Scanning file system metadata, phase 2 ...  
GPFS: 6027-552 Scan completed successfully.  
GPFS: 6027-589 Scanning file system metadata, phase 3 ...  
GPFS: 6027-552 Scan completed successfully.  
GPFS: 6027-589 Scanning file system metadata, phase 4 ...  
GPFS: 6027-552 Scan completed successfully.  
GPFS: 6027-565 Scanning user file metadata ...  
 68 % complete on Thu Feb 16 13:59:06 2006  
100 % complete on Thu Feb 16 13:59:07 2006  
GPFS: 6027-552 Scan completed successfully.  
Done
```

See also

- “mmchdisk command” on page 354
- “mmcrnsd command” on page 426
- “mmdeldisk command” on page 449
- “mmlsdisk command” on page 528
- “mmlsnsd command” on page 549
- “mmlspool command” on page 554

Location

/usr/lpp/mmfs/bin

mmaddnode command

Adds nodes to a GPFS cluster.

Synopsis

```
mmaddnode -N {NodeDesc[,NodeDesc...] | NodeFile}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmaddnode** command to add nodes to an existing GPFS cluster. On each new node, a mount point directory and character mode device is created for each GPFS file system.

Follow these rules when adding nodes to a GPFS cluster:

- You may issue the command only from a node that already belongs to the GPFS cluster.
- While a node may mount file systems from multiple clusters, the node itself may only be added to a single cluster using the **mmcrcluster** or **mmaddnode** command.
- The nodes must be available for the command to be successful. If any of the nodes listed are not available when the command is issued, a message listing those nodes is displayed. You must correct the problem on each node and reissue the command to add those nodes.
- After the nodes are added to the cluster, use the **mmchlicense** command to designate appropriate GPFS licenses to the new nodes.

Parameters

```
-N NodeDesc[,NodeDesc...] | NodeFile
```

Specifies node descriptors, which provide information about nodes to be added to the cluster.

NodeFile

Specifies a file containing a list of node descriptors, one per line, to be added to the cluster.

```
NodeDesc[,NodeDesc...]
```

Specifies the list of nodes and node designations to be added to the GPFS cluster. Node descriptors are defined as:

```
NodeName:NodeDesignations:AdminNodeName
```

where:

NodeName

Specifies the host name or IP address of the node for GPFS daemon-to-daemon communication.

The host name or IP address must refer to the communication adapter over which the GPFS daemons communicate. Aliased interfaces are not allowed. Use the original address or a name that is resolved by the **host** command to that original address. You can specify a node using any of these forms:

- Short host name (for example, h135n01)
- Long, fully-qualified, host name (for example, h135n01.ibm.com)
- IP address (for example, 7.111.12.102). IPv6 addresses must be enclosed in brackets (for example, [2001:192::192:168:115:124]).

Regardless of which form you use, GPFS will resolve the input to a host name and an IP address and will store these in its configuration files. It is expected that those values will not change while the node belongs to the cluster.

mmaddnode

NodeDesignations

An optional, "-" separated list of node roles:

- **manager** | **client** – Indicates whether a node is part of the node pool from which file system managers and token managers can be selected. The default is **client**.
- **quorum** | **nonquorum** – Indicates whether a node is counted as a quorum node. The default is **nonquorum**.

Note: If you are designating a new node as a quorum node, and **adminMode central** is in effect for the cluster, GPFS must be down on all nodes in the cluster. Alternatively, you may choose to add the new nodes as **nonquorum** and once GPFS has been successfully started on the new nodes, you can change their designation to **quorum** using the **mmchnode** command.

AdminNodeName

Specifies an optional field that consists of a node interface name to be used by the administration commands to communicate between nodes. If *AdminNodeName* is not specified, the *nodeName* value is used.

You must provide a *NodeDesc* for each node to be added to the GPFS cluster.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmaddnode** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To add nodes **k164n06** and **k164n07** as **quorum** nodes, designating **k164n06** to be available as a **manager** node, issue this command:

```
mmaddnode -N k164n06:quorum-manager,k164n07:quorum
```

To confirm the addition, issue this command:

```
mmiscluster
```

The system displays information similar to:

```
GPFS cluster information
```

```
=====
```

```
GPFS cluster name:    cluster1.kgn.ibm.com
GPFS cluster id:     680681562214606028
GPFS UID domain:     cluster1.kgn.ibm.com
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:     server-based
```

```
GPFS cluster configuration servers:
```

```
-----
```

```
Primary server:     k164n07.kgn.ibm.com
```

Secondary server: k164n04.kgn.ibm.com

Node	Daemon node name	IP address	Admin node name	Designation
1	k164n04.kgn.ibm.com	198.117.68.68	k164n04.kgn.ibm.com	quorum
2	k164n07.kgn.ibm.com	198.117.68.71	k164n07.kgn.ibm.com	quorum
3	k164n06.kgn.ibm.com	198.117.68.70	k164n06.kgn.ibm.com	quorum-manager

See also

- “mmchconfig command” on page 331
- “mmcrcluster command” on page 403
- “mmchcluster command” on page 327
- “mmdelnode command” on page 460
- “mmlscluster command” on page 524

Location

/usr/lpp/mmfs/bin

mmafmconfig command

Can be used to manage home caching behavior and mapping of gateways and home NFS exported servers.

Synopsis

You can use the **mmafmconfig** command to -

- set up or update mapping for parallel I/O by using add, update, or delete options.
- enable or disable extended attributes/sparse file support from the AFM cache.

```
mmafmconfig {add | update} MapName --export-map ExportServerMap
```

or

```
mmafmconfig delete {MapName | all}
```

or

```
mmafmconfig show [MapName | all]
```

or

```
mmafmconfig {enable | disable} ExportPath
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

You can use this command to configure a home cluster for enabling support of extended attributes /sparse files on AFM cache filesets pointing to this home. You must run the **mmafmconfig enable** command on the home path. Running this command creates the `.afm` directory which contains the control-enabled, `directio.afmctl` file. The **mmafmconfig disable** command removes the `.afm` directory from the home path and subsequently, the cache does not support sparse files and files with extended attributes.

You can also use the **mmafmconfig** command with add, update, delete, or show options on the cache site to manage mapping of gateway node with home NFS servers for parallel I/O.

Relink filesets after running this command.

Parameters

MapName

Specifies the name that uniquely identifies the mapping of the gateway nodes with the home NFS exported servers.

--export-map *ExportServerMap*

Specifies a comma-separated list of pairs of home NFS exported server nodes (*ExportServer*) and gateway nodes (*GatewayNode*), in the following format:

```
[ExportServer/GatewayNode] [,ExportServer/GatewayNode] [,...]
```

where:

ExportServer

Is the IP address or host name of a member node in the home cluster *MapName*.

GatewayNode

Specifies a gateway node in the cache cluster (the cluster where the command is issued).

enable

Enables extended attributes or sparse files functions on the AFM cache. Run at the home cluster only.

disable

Disables extended attributes or sparse files functions on the AFM cache. Run at the home cluster only.

ExportPath

Specifies the root of the home exported directory for enabling or disabling the AFM features.

add

Sets up maps for parallel I/O. Run at cache only.

delete

Deletes maps for parallel I/O. Run at cache only.

update

Updates maps for parallel I/O. Run at cache only.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmafmconfig** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See “Requirements for administering a GPFS file system” on page 1 in *IBM Spectrum Scale: Administration and Programming Reference*

Example

The following is an example of a mapping for an NFS target, assuming four cache gateway nodes hs22n18, hs22n19, hs22n20, and hs22n21, mapped to two home NFS servers js22n01 and js22n02 (192.168.200.11 and 192.168.200.12) and then creating single writer filesets by using the following mapping:

1. Issue the following command:

```
# mmafconfig add mapping1 --export-map js22n01/hs22n18,js22n02/hs22n19
```

The system displays output similar to: mmafconfig: Command successfully completed.

- 2.

Issue the following command:

```
# mmafconfig add mapping2 --export-map js22n02/hs22n20,js22n01/hs22n21
```

The system displays output similar to: mmafconfig: Command successfully completed.

mmafconfig: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

- 3.

Issue the following command:

```
# mmafconfig show
```

The system displays output similar to: Map name: mapping1

Export server map: 192.168.200.12/hs22n19.gpfs.net,192.168.200.11/hs22n18.gpfs.net

mmafmconfig

Map name: mapping2

Export server map: 192.168.200.11/hs22n20.gpfs.net,192.168.200.12/hs22n21.gpfs.net

4.

#Create filesets using these mappings:

Issue the following commands: **mmcrfileset gpfs1 sw1 -inode-space new -p afmmode=sw,afmtarget=mapping1://gpfs/gpfs2/swhome**

mmcrfileset gpfs1 ro1 -inode-space new -p afmmode=ro,afmtarget=mapping2://gpfs/gpfs2/swhome

See also

- “mmafmctl command” on page 251
- “mmafmlocal command” on page 264
- “mmchconfig command” on page 331
- “mmchfileset command” on page 365
- “mmchfs command” on page 371
- “mmcrfileset command” on page 408
- “mmcrfs command” on page 414
- “mmlsconfig command” on page 526
- “mmlsfileset command” on page 532
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmafmctl command

This command is for various operations and reporting information on all filesets. It is recommended to read the *IBM Spectrum Scale: Advanced Administration Guide* AFM and AFM Disaster Recovery chapters in conjunction with this manual for detailed description of the functions.

Synopsis

To use the AFM Disaster Recovery functions correctly, it is strongly recommended to use all commands enlisted in this chapter in accordance with the steps described in the AFM Disaster Recovery chapter in *IBM Spectrum Scale: Advanced Administration Guide*.

AFM read only mode is referred as RO, single writer mode is referred as SW, independent writer mode is referred as IW and local update mode is referred as LU in this manual.

```
mmafmctl Device {resync | expire | unexpire} -j FilesetName
```

or

```
mmafmctl Device {getstate | resumeQueued} [-j FilesetName]
```

or

```
mmafmctl Device flushPending [-j FilesetName [--list-file ListFile]]
[-s LocalWorkDirectory]
```

or

```
mmafmctl Device failover -j FilesetName
--new-target NewAfmTarget [--target-only] [-s LocalWorkDirectory]
```

or

```
mmafmctl Device prefetch -j FilesetName [--metadata-only]
[ [--list-file ListFile] |
  { --home-list-file HomeListFile |
    { --home-inode-file PolicyListFile } }
  [--home-fs-path HomeFileSystemPath]
  [-s LocalWorkDirectory]
```

or

```
mmafmctl Device evict -j FilesetName
[ --safe-limit SafeLimit ] [--order {LRU | SIZE}]
[ --log-file LogFile ] [--filter Attribute=Value ...]
```

or

```
mmafmctl Device failback -j FilesetName { --start --failover-time Time } | --stop }
[-sLocalWorkDirectory]
```

or

```
mmafmctl Device failoverToSecondary -j FilesetName [--norestore | --restore ]
```

or

```
mmafmctl Device convertToPrimary -j FilesetName
[ --afmtarget Target { --inband | --outband | --secondary-snapname SnapshotName } ]
[ --check-metadata | --nocheck-metadata ] [--rpo RPO] [-s LocalWorkDirectory]
```

or

```
mmafmctl Device convertToSecondary -j FilesetName --primaryid PrimaryId [ --force ]
```

or

mmafmctl

```
mmafmctl Device changeSecondary -j FilesetName  
--new-target NewAfmTarget [ --target-only | --inband | --outband ]  
[-s LocalWorkDirectory]
```

or

```
mmafmctl Device replacePrimary -j FilesetName
```

or

```
mmafmctl Device failbackToPrimary -j FilesetName {--start | --stop [ --force ] }
```

or

```
mmafmctl Device {applyUpdates |getPrimaryId } -j FilesetName
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The usage of options of this command for different operations on both AFM (RO/SW/IW/LU) filesets and AFM primary/secondary filesets are explained with examples.

File system should be mounted on all gateway nodes for **mmafmctl** functions to work.

Parameters

Device

Specifies the device name of the file system.

-j *FilesetName*

Specifies the fileset name.

-s *LocalWorkDirectory*

Specifies the temporary working directory.

1. This section describes:

```
mmafmctl Device {resync | expire | unexpire} -j FilesetName
```

resync

This option is available only for SW cache. In case of inadvertent changes made at home of an SW fileset, such as delete of a file or change of data in a file etc., the administrator can correct the home by sending all contents from cache to home using this option. The limitation of this option that renamed files at home may not be fixed by **resync**. Using **resync** requires the cache to be either in NeedsResync or Active state.

expire | unexpire

This option is available only for RO cache. When an RO cache is disconnected, the cached contents are still accessible for the user. However the administrator can define a time from home beyond which access to the cached contents becomes stale. Such an event would occur automatically after disconnection (when cached contents are no longer accessible) and is called *expiration*; the cache is said to be expired. This state can also be forced manually using the **expire** parameter.

When the home comes back or reconnects, the cache contents become automatically accessible again and the cache is said to un-expire. This can be forced manually using the **unexpire** parameter.

The manual expiration and un-expiration can be forced on a cache even when the home is in a connected state. For expiring a fileset manually the **afmExpirationTimeout** needs to have been set on the fileset. If a cache is expired using this manual method, it will also have to be manually unexpired.

2. This section describes:

```
mmafmctl Device {getstate | resumeRequeued} [-j FilesetName]
```

getstate

This option is applicable for all AFM (RO/SW/IW/LU) and AFM primary filesets. It displays the status of the fileset in the following fields:

Fileset Name

The name of the fileset.

Fileset Target

The host server and the exported path on it.

Gateway Node

Metadata server or MDS of the fileset. This gateway node is handling requests for this fileset.

Queue Length

Current length of the queue on the MDS.

Queue numExec

Number of operations played at home since the fileset is last Active.

Cache State

- Cache states applicable for all AFM RO/SW/IW/LU filesets:
Active, Inactive, Dirty, Disconnected, Unmounted
- Cache states applicable for RO filesets:
Expired
- Cache states applicable for SW and IW filesets:
Recovery, FlushOnly, QueueOnly, Dropped, NeedsResync, FailoverInProgress
- Cache states applicable for IW filesets:
FailbackInProgress, FailbackCompleted, NeedsFailback
- Cache states applicable for AFM primary filesets:
PrimInitInProg, PrimInitFail, Active, Inactive, Dirty, Disconnected, Unmounted,
FailbackInProg, Recovery, FlushOnly, QueueOnly, Dropped, NeedsResync

All cache states are explained in AFM and AFM Disaster Recovery chapters in the *IBM Spectrum Scale: Advanced Administration Guide*. Please refer to them for more information.

resumeRequeued

This option is applicable for SW/IW and primary filesets. If there are operations in the queue that were re-queued due to errors at home, the Administrator should correct those errors and can run this option to retry the re-queued operations.

3. This section describes:

```
mmafmctl Device flushPending [-j FilesetName [--list-file ListFile]]
[-s LocalWorkDirectory]
```

flushPending

Flushes all point-in-time pending messages in the normal queue on the fileset to home. Requeued messages and messages in the priority queue for the fileset are not flushed by this command.

When **--list-file** *ListFile* is specified, the messages pending on the files listed in the list file are flushed to home.

4. This section describes:

```
mmafmctl Device failover -j FilesetName
--new-target NewAfmTarget [--target-only] [-s LocalWorkDirectory]
```

mmafmctl

This option is applicable only for SW/IW filesets. This option pushes all the data from cache to home. It should be used only in case home is completely lost due to a disaster and a new home is being set up. Failover often takes a long time to complete; status can be checked using the `afmManualResyncComplete` callback or via `mmafmctl getstate` command.

--new-target *NewAfmTarget*

Specifies a new home server and path, replacing the home server and path originally set by the `afmTarget` parameter of the `mmcrfileset` command. Specified in either of the following formats:

Protocol://[Host|Map]/Path

or

{Host|Map}:Path

where:

Protocol://

Specifies the transport protocol. Valid values are `nfs://` or `gpfs://`.

Host|Map

Host

Specifies the server domain name system (DNS) name or IP address.

Map

Specifies the export map name.

Notes:

1. When specifying `nfs://` as the value for *Protocol://*, you must provide a value for *Host* or *Map*.
2. When specifying `gpfs://` as the value for *Protocol://*, do not provide a value for *Host*. However, provide a value for *Map* if it refers to an export map entry.

Path

Specifies the export path.

It is possible to change the protocol along with the target using failover. For example, a cache using an NFS target `bear110:/gpfs/gpfsA/home` can be switched to a GPFS target whose remote file system is mounted at `/gpfs/fs1`, and vice-versa, as follows:

```
mmafmctl fs0 failover -j afm-mc1 --new-target gpfs:///gpfs/fs1
mmafmctl fs0 failover -j afm-mc1 --new-target nfs://bear110/gpfs/gpfsA/home
```

Note that in the first command, `///` is needed because *Host* is not provided.

--target-only

This is used if the user wants to change the mount path/IP address in the target path. The new NFS server should be in the same home cluster and should be of the same architecture as the existing NFS server in the target path. This option should not be used to change the target location or protocol.

5. This section describes:

```
mmafmctl Device prefetch -j FilesetName [--metadata-only]
    [--list-file ListFile] |
    [--home-fs-path HomeFileSystemPath]
    [-s LocalWorkDirectory]
```

This option is used for fetching file contents from home before the application requests for the contents. This reduces the network delay when the application is in progress. You can also use this option to move files over the WAN when the WAN usage is low. These files might be the files that are accessed during high WAN usage. Thus, you can use this option for better WAN management.

You can use the prefetch option to -

- populate data
- populate metadata
- view prefetch statistics

If you run **prefetch** without providing any options, it displays statistics of the last **prefetch** command run on the fileset.

Prefetch completion can be monitored using the **afmPrepopEnd** event.

--metadata-only

Prefetches only the metadata and not the actual data. This is useful in migration scenarios. This option requires the list of files whose metadata you want. Hence it must be combined with a list file option.

--list-file *ListFile*

The specified file is a file containing a list of files, and needs to be prefetched, one file per line. All files must have fully qualified path names.

If the list of files to be prefetched have filenames with special characters then a policy must be used to generate the listfile. Remove entries from the file other than the filenames.

An indicative list of files:

- files with fully qualified names from cache
- files with fully qualified names from home
- list of files from home generated using policy. Do not edit.

--home-list-file *HomeListFile*

Contains a list of files from the home cluster that needs to be prefetched, one file per line. All files must have fully qualified path names. If the list of files has filenames with special characters, use a policy to generate the listfile. Edit to remove all entries other than the filenames.

This command is deprecated. Use **-list-file** instead.

--home-inode-file *PolicyListFile*

Contains a list of files from the home cluster that needs to be prefetched in the cache. Do not edit the file. The file is generated using policy.

This command is deprecated. Use **-list-file** instead.

--home-fs-path *HomeFileSystemPath*

Specifies the full path to the fileset at the home cluster and can be used in conjunction with *ListFile*.

You must use this option when the mount point on the gateway nodes of the *afmTarget* filesets does not match with the mount point on the GPFS home cluster.

For example, **mmafmctl gpfs1 prefetch -j cache1 -list-file /tmp/list.allfiles --home-fs-path /gpfs/remotefs1**

In this example, the file system is mounted on the :

- home cluster at /gpfs/homefs1
- gateway nodes at /gpfs/remotefs1

Prefetch is an asynchronous process and you can use the fileset when prefetch is in progress. You can monitor Prefetch using the **afmPrepopEnd** event. AFM can prefetch the data using the **mmafmctl prefetch** command (which specifies a list of files to prefetch). Prefetch always pulls the complete file contents from home and AFM automatically sets a file as cached when it is completely prefetched.

6. This section describes:

mmafctl

```
mmafctl Device evict -j FilesetName  
      [--safe-limit SafeLimit] [--order {LRU | SIZE}]  
      [--log-file LogFile] [--filter Attribute=Value ...]
```

This option is applicable for RO/SW/IW/LU filesets. When cache space exceeds the allocated quota, data blocks from non-dirty are automatically de-allocated with the eviction process. This option can be used for a file that is specifically to be de-allocated based on some criteria. All options can be combined with each other.

--safe-limit *SafeLimit*

This is a compulsory parameter for the manual evict option, for order and filter attributes. Specifies target quota limit (which is used as the low water mark) for eviction in bytes; must be less than the soft limit. This parameter can be used alone or can be combined with one of the following parameters (order or filter attributes). Specify the parameter in bytes.

--order LRU | SIZE

Specifies the order in which files are to be chosen for eviction:

LRU

Least recently used files are to be evicted first.

SIZE

Larger-sized files are to be evicted first.

--log-file *LogFile*

Specifies the file where the eviction log is to be stored. The default is that no logs are generated.

--filter *Attribute=Value*

Specifies attributes that enable you to control how data is evicted from the cache. Valid attributes are:

FILENAME=*FileName*

Specifies the name of a file to be evicted from the cache. This uses an SQL-type search query. If the same file name exists in more than one directory, it will evict all the files with that name. The complete path to the file should not be given here.

MINFILESIZE=*Size*

Sets the minimum size of a file to evict from the cache. This value is compared to the number of blocks allocated to a file (**KB_ALLOCATED**), which may differ slightly from the file size.

MAXFILESIZE=*Size*

Sets the maximum size of a file to evict from the cache. This value is compared to the number of blocks allocated to a file (**KB_ALLOCATED**), which may differ slightly from the file size.

Possible combinations of *safelimit*, *order*, and *filter* are:

- only Safe limit
- Safe limit + LRU
- Safe limit + SIZE
- Safe limit + FILENAME
- Safe limit + MINFILESIZE
- Safe limit + MAXFILESIZE
- Safe limit + LRU + FILENAME
- Safe limit + LRU + MINFILESIZE
- Safe limit + LRU + MAXFILESIZE
- Safe limit + SIZE + FILENAME
- Safe limit + SIZE + MINFILESIZE
- Safe limit + SIZE + MAXFILESIZE

7. This section describes:

```
mmafctl Device failback -j FilesetName [--start --failover-time Time] | --stop  
      [-sLocalWorkDirectory]
```

failback is applicable only for IW filesets.

failback --start --failover-time *Time*

Specifies the point in time at the home cluster, from which the independent-writer cache taking over as writer should sync up. *Time* can be specified in date command format with time zones. It will use the cluster's time-zone and year by default.

failback --stop

An option to be run after the failback process is complete and the fileset moves to **FailbackCompleted** state. This will move the fileset to **Active** state.

8. This section describes:

```
mmafmctl Device failoverToSecondary -j FilesetName [--norestore | --restore ]
```

This is to be run on a secondary fileset.

When primary experiences a disaster, all applications will need to be moved to the secondary to ensure business continuity. The secondary has to be first converted to an acting primary using this option.

There is a choice of restoring the latest snapshot data on the secondary during the failover process or leave the data as is using the **--norestore** option. Once this is complete, the secondary becomes ready to host applications.

--norestore

Specifies that restoring from the latest peer snapshot is not required.

--restore

Specifies that the restoring of data is to be done from the latest peer snapshot. This is the default.

9. This section describes:

```
mmafmctl Device convertToPrimary -j FilesetName
[ --afmtarget Target { --inband | --outband | --secondary-snapname SnapshotName } ]
[ --check-metadata | --nocheck-metadata ] [--rpo RPO] [-s LocalWorkDirectory]
```

This is to be run on a GPFS fileset or SW/IW fileset which is intended to be converted to primary.

--afmtarget *Target*

Specifies the secondary that needs to be configured for this primary. Need not be used for AFM filesets as target would already have been defined.

--inband

Used for inband trucking. *Inband trucking* is copying the data while setting up a primary/secondary relationship from GPFS filesets, where primary site has contents and secondary site is empty.

--outband

Used for outband trucking. *Outband trucking* is copying data manually using other ways such as **ftp**, **scp**, **rsync** etc. This should be completed before the relationship is established.

--check-metadata

This checks if the disallowed types (like immutable/append-only files) are present in the GPFS fileset on the primary site before the conversion. Conversion with this option fails if such files exist. For SW/IW filesets, presence of orphans and incomplete directories are also checked. SW/IW filesets should have established contact with at least once home for this option to succeed.

--nocheck-metadata

Used if one needs to proceed with conversion without checking for appendonly/immutable files.

--secondary-snapname *SnapshotName*

Used while establishing a new primary for an existing secondary or acting primary during failback.

mmafctl

--rpo *RPO*

Specifies the RPO interval in minutes for this primary fileset.

10. This section describes:

```
mmafctl Device convertToSecondary -j FilesetName --primaryid PrimaryId [ --force ]
```

This is to be run on a GPFS fileset on the secondary site. This converts a GPFS independent fileset to a secondary and sets the primary ID.

--primaryid *PrimaryId*

Specifies the ID of the primary with which the secondary will be associated.

--force

If **convertToSecondary** failed or got interrupted, it will not create afmctl file at the secondary. The command should be rerun with the **--force** option.

11. This section describes:

```
mmafctl Device changeSecondary -j FilesetName  
--new-target NewAfmTarget [ --target-only | --inband | --outband ]  
[-s LocalWorkDirectory]
```

This is to be run on a primary fileset only.

A disaster at the secondary can take place due to which secondary is not available.

Run this command on the primary when a secondary fails and this primary needs to be connected with a new secondary. On the new secondary site a new GPFS independent fileset has to be created. Data on the primary can be copied to the new GPFS fileset that was created with this command using other means such as **ftp**, **scp** etc. Alternatively it can be decided that data will be trucked using the relationship.

--new-target *NewAfmTarget*

Used to mention the new secondary.

--inband | **--outband**

Used based on the method used to truck data.

--target-only

Used when you want to change the IP address or NFS server name for the same target path. The new NFS server must be in the same home cluster and must be of the same architecture(power or x86) as the existing NFS server in the target path. This option can be used to move from NFS to a mapping target.

12. This section describes:

```
mmafctl Device replacePrimary -j FilesetName
```

This is used on an acting primary only. This will create a latest snapshot of the acting primary. This command deletes any old peer snapshots on the acting primary and creates a new initial peer snapshot psnap0.

This snapshot will be used in the setup of the new primary.

13. This section describes:

```
mmafctl Device fallbackToPrimary -j FilesetName {--start | --stop [ --force ] }
```

This is to be run on an old primary that came back after the disaster, or on a new primary that is to be configured after an old primary went down with a disaster. The new primary should have been converted from GPFS to primary using **convertToPrimary** option.

--start

Restores the primary to the contents from the last RPO on the primary before the disaster. This option will put the primary in read-only mode, to avoid accidental corruption until the failback process is completed. In case of new primary that is setup using **convertToPrimary**, the **failback --start** does no change.

--stop

Used to complete the Failback process. This will put the fileset in read-write mode. The primary is now ready for starting applications.

--force

Used if **--stop** option does not complete due to errors and not allow for failback to be stopped.

14. This section describes:

```
mmafmctl Device {applyUpdates |getPrimaryId } -j FilesetName
```

Both options are intended for the primary fileset.

applyUpdates

Run this on the primary after running the **failback --start** command. All the differences can be brought over in one go or through multiple iterations. For minimizing application downtime, this command can be run multiple times to bring the primary's contents in sync with the acting primary. When the contents are as close as possible or minimal, applications should take a downtime and then this command should be run one last time.

It is possible that **applyUpdates** fails with an error during instances when the acting primary is overloaded. In such cases the command has to be run again.

getPrimaryID

Used to get primary Id of a primary fileset.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmafmctl** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See the following *IBM Spectrum Scale: Administration and Programming Reference* topic: "Requirements for administering a GPFS file system" on page 1.

Examples**1. running resync on SW:**

```
# mmafctl fs1 resync -j sw1
mmafctl: Performing resync of fileset: sw1
```

```
# mmafctl fs1 getstate -j sw1
Fileset Name Fileset Target
```

Fileset Name	Fileset	Target	Cache State	Gateway Node	Queue Length	Queue numExec
sw1	nfs://c26c3apv2/gpfs/homefs1/newdir1		Dirty	c26c2apv1	4067	10844

2. Expiring a RO fileset:

```
# mmafctl fs1 expire -j rol
```

```
# mmafctl fs1 getstate -j rol
```

mmafmctl

Fileset Name	Fileset Target	Cache State	Gateway Node	Queue Length	Queue numExec
rol	gpfs:///gpfs/remotefs1/dir1	Expired	c26c4apv1	0	4

3. Unexpiring a RO fileset:

```
# mmafmctl fs1 unexpire -j rol
```

```
# mmafmctl fs1 getstate -j rol
```

Fileset Name	Fileset Target	Cache State	Gateway Node	Queue Length	Queue numExec
rol	gpfs:///gpfs/remotefs1/dir1	Active	c26c4apv1	0	4

4. Run flushPending on SW fileset:

```
// Populate the fileset with data
```

```
# mmafmctl fs1 getstate -j sw1
```

Fileset Name	Fileset Target	Cache State	Gateway Node	Queue Length	Queue numExec
sw1	gpfs:///gpfs/remotefs1/dir1	Dirty	c26c2apv1	5671	293

Get the list of files newly created using policy:

```
RULE EXTERNAL LIST 'L' RULE 'List' LIST 'L' WHERE PATH_NAME LIKE '%'
```

```
# mmapplypolicy /gpfs/fs1/sw1/migrateDir.popFSDir.22655 -P p1 -f p1.res -L 1 -N mount -I defer
```

```
Policy created this file, this should be hand-edited to retain only the names:  
11012030 65537 0 -- /gpfs/fs1/sw1/migrateDir.popFSDir.22655/file_with_posix_acl1  
11012032 65537 0 -- /gpfs/fs1/sw1/migrateDir.popFSDir.22655/populateFS.log  
11012033 65537 0 --  
/gpfs/fs1/sw1/migrateDir.popFSDir.22655/sparse_file_0_with_0_levels_indirection
```

```
# cat p1.res.list | awk '{print $5}' > /lfile
```

```
# mmafmctl fs1 flushPending -j sw1 --list-file=/lfile
```

5. Failover of SW to a new home:

```
# mmafmctl fs1 getstate -j sw1
```

Fileset Name	Fileset Target	Cache State	Gateway Node	Queue Length	Queue numExec
sw1	gpfs:///gpfs/remotefs1/dir1	Dirty	c26c2apv1	785	5179

```
# mmcrfileset homefs1 newdir1 --inode-space=new  
Fileset newdir1 created with id 219 root inode 52953091.
```

```
# mmlinkfileset homefs1 newdir1 -J /gpfs/homefs1/newdir1  
Fileset newdir1 linked at /gpfs/homefs1/newdir1
```

```
# mmafmconfig /gpfs/homefs1/newdir1 enable
```

```
# mmafmctl fs1 failover -j sw1 --new-target=c26c3apv1:/gpfs/homefs1/newdir1  
mmafmctl: Performing failover to nfs://c26c3apv1/gpfs/homefs1/newdir1  
Fileset sw1 changed.  
mmafmctl: Failover in progress. This may take while...  
Check fileset state or register for callback to know the completion status.
```

Callback registered, logged into mmfs.log:

```
Thu May 21 03:06:18.303 2015: [I] Calling User Exit Script callback7: event  
afmManualResyncComplete, Async command recovery.sh
```

```
# mmafmctl fs1 getstate -j sw1
```

Fileset Name	Fileset Target	Cache State	Gateway Node	Queue Length	Queue numExec
sw1	nfs://c26c3apv1/gpfs/homefs1/newdir1	Active	c26c2apv1	0	5250

6. Changing target of SW fileset:

Changing to another NFS server in the same home cluster using --target-only option:

```
# mmafmctl fs1 failover -j sw1 --new-target=c26c3apv2:/gpfs/homefs1/newdir1 --target-only
mmafmctl: Performing failover to nfs://c26c3apv2/gpfs/homefs1/newdir1
Fileset sw1 changed.
```

```
# mmafmctl fs1 getstate -j sw1
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
-----
sw1 nfs://c26c3apv2/gpfs/homefs1/newdir1 Active c26c2apv1 0 5005
```

7. metadata population using prefetch:

```
# mmafmctl fs1 getstate -j ro
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
-----
ro nfs://c26c3apv1/gpfs/homefs1/dir3 Active c26c2apv2 0 7
```

List Policy:

```
RULE EXTERNAL LIST 'List' RULE 'List' LIST 'List' WHERE PATH_NAME LIKE '%'
```

Run the policy at home:

```
mmapplypolicy /gpfs/homefs1/dir3 -P px -f px.res -L 1 -N mount -I defer
```

Policy created this file, this should be hand-edited to retain only file names.

This file can be used at the cache to populate metadata.

```
# mmafmctl fs1 prefetch -j ro --metadata-only -list-file=px.res.list.List
mmafmctl: Performing prefetching of fileset: ro
```

Prefetch end can be monitored using this event:

```
Thu May 21 06:49:34.748 2015: [I] Calling User Exit Script prepop: event afmPrepopEnd,
Async command prepop.sh.
```

The statistics of the last prefetch command is viewed by:

```
mmafmctl fs1 prefetch -j ro
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async Read (Total) Async Read (Data in Bytes)
-----
ro 0 1 0 7 0
```

8. Prefetch of data using --home-list-file option:

```
# cat /lfile1
/gpfs/homefs1/dir3/file1
/gpfs/homefs1/dir3/dir1/file1
# mmafmctl fs1 prefetch -j ro --home-list-file=/lfile1
mmafmctl: Performing prefetching of fileset: ro
```

```
# mmafmctl fs1 prefetch -j ro
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async Read (Total) Async Read (Data in Bytes)
-----
ro 0 0 0 2 122880
```

9. Prefetch of data using --home-inode-file option:

Inode file is created using the above policy at home, and should be used as such without hand-editing.

List Policy:

```
RULE EXTERNAL LIST 'List' RULE 'List' LIST 'List' WHERE PATH_NAME LIKE '%'
```

Run the policy at home:

```
# mmapplypolicy /gpfs/homefs1/dir3 -P px -f px.res -L 1 -N mount -I defer
```

```
# cat /lfile2
```

mmafmctl

```
113289 65538 0 -- /gpfs/homefs1/dir3/file2
113292 65538 0 -- /gpfs/homefs1/dir3/dir1/file2
# mmafmctl fs1 prefetch -j ro --home-inode-file=/lfile2
mmafmctl: Performing prefetching of fileset: ro
```

```
mmafmctl fs1 prefetch -j ro
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async Read (Total) Async Read (Data in Bytes)
-----
ro                0                0                2                2                0
```

10. Using --home-fs-path option for a target with NSD protocol:

```
# mmafmctl fs1 getstate -j ro2
Fileset Name Fileset Target          Cache State Gateway Node Queue Length Queue numExec
-----
ro2          gpfs:///gpfs/remotefs1/dir3 Active   c26c4apv1  0          7
```

```
# cat /lfile2
113289 65538 0 -- /gpfs/homefs1/dir3/file2
113292 65538 0 -- /gpfs/homefs1/dir3/dir1/file2
# mmafmctl fs1 prefetch -j ro2 --home-inode-file=/lfile2 --home-fs-path=/gpfs/homefs1/dir3
mmafmctl: Performing prefetching of fileset: ro2
```

```
# mmafmctl fs1 prefetch -j ro2
Fileset Name Async Read (Pending) Async Read (Failed) Async Read (Already Cached) Async Read (Total) Async Read (Data in Bytes)
-----
ro2                0                0                0                2            122880
```

11. Manually evicting using safe-limit and filename parameters:

```
# ls -lis /gpfs/fs1/ro2/file10M_1
12605961 10240 -rw-r--r-- 1 root root 10485760 May 21 07:44 /gpfs/fs1/ro2/file10M_1
```

```
# mmafmctl fs1 evict -j ro2 --safe-limit=1 --filter FILENAME=file10M_1
```

```
# ls -lis /gpfs/fs1/ro2/file10M_1
12605961 0 -rw-r--r-- 1 root root 10485760 May 21 07:44 /gpfs/fs1/ro2/file10M_1
```

12. IW Failback:

```
# mmafmctl fs1 getstate -j iw1
Fileset Name Fileset Target          Cache State Gateway Node Queue Length Queue numExec
-----
iw1          nfs://c26c3apv1/gpfs/homefs1/dir3 Active   c25m4n03  0          8
```

```
# touch file3 file4
```

```
# mmafmctl fs1 getstate -j iw1
Fileset Name Fileset Target          Cache State Gateway Node Queue Length Queue numExec
-----
iw1          nfs://c26c3apv1/gpfs/homefs1/dir3 Dirty    c25m4n03  2          11
```

```
Unlink IW fileset feigning failure:
# mmunlinkfileset fs1 iw1 -f
Fileset iw1 unlinked.
```

```
Write from IW home, assuming applications failed over to home:
Thu May 21 08:20:41 4]dir3# touch file5 file6
Relink IW back on the cache cluster, assuming it came back up:
# mmlinkfileset fs1 iw1 -J /gpfs/fs1/iw1
Fileset iw1 linked at /gpfs/fs1/iw1
```

```
Run failback on IW:
# mmafmctl fs1 failback -j iw1 --start --failover-time='May 21 08:20:41'
```

```
# mmafmctl fs1 getstate -j iw1
Fileset Name Fileset Target          Cache State Gateway Node Queue Length Queue numExec
-----
iw1          nfs://c26c3apv1/gpfs/homefs1/dir3 FailbackInProg c25m4n03  0          0
```

```
# mmafmctl fs1 failback -j iw1 -stop
```

```
# mmafmctl fsl getstate -j iw1
Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
-----
iw1          nfs://c26c3apv1/gpfs/homefs1/dir3 Active      c25m4n03   0           3
```

See also

- “mmafmconfig command” on page 248
- “mmafmlocal command” on page 264
- “mmchattr command” on page 321
- “mmchconfig command” on page 331
- “mmchfileset command” on page 365
- “mmchfs command” on page 371
- “mmcrfileset command” on page 408
- “mmcrfs command” on page 414
- “mmlsconfig command” on page 526
- “mmlsfileset command” on page 532
- “mmlsfs command” on page 536
- “mmpsnap command” on page 611

See also the following *IBM Spectrum Scale: Advanced Administration Guide* topics:

- The chapter about AFM, for detailed descriptions of the AFM functions
- The chapter about AFM disaster recovery, for detailed descriptions of the AFM disaster recovery functions.

Location

/usr/lpp/mmfs/bin

mmafmlocal command

Provides a list of cached files and file statistics such as inode number, allocated blocks, and so on.

Synopsis

```
mmafmlocal ls [FileName ...]
```

or

```
mmafmlocal stat FileName ...
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmafmlocal** command provides information about files that exist in the cache.

Parameters

ls Lists files with data that is in the cache already. This parameter is valid for fully-cached files only.

FileName

Specifies the name of a file to be listed.

stat

Displays statistics for files. If the file is not cached already, the number of allocated blocks is zero. This parameter is valid for partially-cached and fully-cached files.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmafmlocal** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

See also

- “mmafmconfig command” on page 248
- “mmafmctl command” on page 251
- “mmchattr command” on page 321
- “mmchconfig command” on page 331
- “mmchfileset command” on page 365
- “mmchfs command” on page 371
- “mmcrfileset command” on page 408
- “mmcrfs command” on page 414
- “mmlsconfig command” on page 526
- “mmlsfileset command” on page 532

- “mmlsfs command” on page 536
- “mmpsnap command” on page 611

Location

/usr/lpp/mmfs/bin

mmapplypolicy command

Deletes files, migrates files between storage pools, or does file compression or decompression in a file system as directed by policy rules.

Synopsis

```
mmapplypolicy {Device|Directory}
[-A IscanBuckets] [-a IscanThreads] [-B MaxFiles]
[-D yyyy-mm-dd[@hh:mm[:ss]]] [-e] [-f FileListPrefix]
[-g GlobalWorkDirectory] [-I {yes|defer|test|prepare}]
[-i InputFileList] [-L n] [-M name=value...] [-m ThreadLevel]
[-N {all | mount | Node[,Node...] | NodeFile | NodeClass}]
[-n DirThreadLevel] [-P PolicyFile] [-q] [-r FileListPathname...]
[-S SnapshotName] [-s LocalWorkDirectory]
[--choice-algorithm {best | exact | fast}]
[--max-merge-files] [--max-sort-bytes]
[--other-sort-options SortOptions] [--qos QOSClass]
[--scope {filesystem | inodespace | fileset}]
[--single-instance] [--sort-buffer-size Size]
[--sort-command SortCommand] [--split-filelists-by-weight]
[--split-margin n.n]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

You can use the **mmapplypolicy** command to apply rules that manage the following types of tasks:

- Migration and replication of file data to and from storage pools.
- Deleting files.
- File compression or decompression. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration and Programming Reference*.

For more information about policy rules, see the topic *Policies for automating file management* in the *IBM Spectrum Scale: Advanced Administration Guide*.

You can run the **mmapplypolicy** command from any node in the cluster that has mounted the file system.

The **mmapplypolicy** command does not affect placement rules (for example, the **SET POOL** and **RESTORE** rule) that are installed for a file system by the **mmchpolicy** command. To display the currently installed rules, issue the **mmlspolicy** command.

A given file can match more than one list rule, but will be included in a given list only once. *ListName* provides the binding to an **EXTERNAL LIST** rule that specifies the executable program to use when processing the generated list.

The **EXTERNAL POOL** rule defines an external storage pool. This rule does not match files, but serves to define the binding between the policy language and the external storage manager that implements the external storage.

Any given file is a potential candidate for at most one **MIGRATE** or **DELETE** operation during one invocation of the **mmapplypolicy** command. That same file may also match the first applicable **LIST** rule.

A file that matches an **EXCLUDE** rule is not subject to any subsequent **MIGRATE**, **DELETE**, or **LIST** rules. You should carefully consider the order of rules within a policy to avoid unintended consequences.

For detailed information on GPFS policies, see the *IBM Spectrum Scale: Advanced Administration Guide*.

This command cannot be run from a Windows node. The GPFS API, documented functions in **gpfs.h** are not implemented on Windows, however the policy language does support the Windows file attributes, so you can manage your GPFS Windows files using the **mmapplypolicy** command running on an AIX or Linux node.

Note: To terminate **mmapplypolicy**, use the **kill** command to send a **SIGTERM** signal to the process group running **mmapplypolicy**.

For example, on Linux if you wanted to terminate **mmapplypolicy** on a process group whose ID is 3813, you would enter the following:

```
kill -s SIGTERM -- -3813
```

If you need to determine which process group is running **mmapplypolicy**, you can use the following command (which also tells you which process groups are running **tsapolicy** and **mmhelp-apolicy**):

```
mmdsh -N all ps auxw | grep policy
```

The system displays output similar to the following:

```
root 31666 0.0 0.0 84604 2328 ? S1 07:29 0:00 /usr/lpp/mmfs/bin/mmhelp-apolicy na -X 10.222.4.12 -s /tmp -Y -x 36845 -m 24 -n 24 -a 2 -L 1 -d 00 -z 15
root 3813 0.3 0.1 68144 4792 pts/1 S 07:29 0:00 /bin/ksh /usr/lpp/mmfs/bin/mmapplypolicy /mak/millions -P /ghome/makaplan/policies/lp.policy -N all
root 3847 127 0.1 455228 5808 pts/1 S1 07:29 0:38 /usr/lpp/mmfs/bin/tsapolicy /mak/millions -P /ghome/makaplan/policies/lp.policy -l yes -L 1 -X 10.222.4.12 -N all
root 3850 0.0 0.0 84832 1620 pts/1 S1 07:29 0:00 /usr/lpp/mmfs/bin/tsapolicy /mak/millions -P /ghome/makaplan/policies/lp.policy -l yes -L 1 -X 10.222.4.12 -N all
root 3891 0.0 0.0 61156 768 pts/2 S+ 07:29 0:00 grep policy
```

Parameters

Device

Specifies the device name of the file system from which files will have the policy rules applied. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**. If specified, this must be the first parameter.

Directory

Specifies the fully-qualified path name of a GPFS file system subtree from which files will have the policy rules applied. If specified, this must be the first parameter.

-A *IsScanBuckets*

Specifies the number of buckets of inode numbers (number of inode/filelists) to be created by the parallel directory scan and processed by the parallel inode scan. Affects the execution of the high-performance protocol that is used when both **-g** and **-N** are specified.

Tip: Set this parameter to the expected number of files to be scanned divided by one million. Then each bucket will have about one million files.

-a *IsScanThreads*

Specifies the number of threads and sort pipelines each node will run during the parallel inode scan and policy evaluation. It affects the execution of the high-performance protocol that is used when both **-g** and **-N** are specified. The default is 2. Using a moderately larger number can significantly improve performance, but might "strain" the resources of the node. In some environments a large value for this parameter can lead to a command failure.

Tip: Set this parameter to the number of CPU "cores" implemented on a typical node in your GPFS cluster.

-B *MaxFiles*

Specifies how many files are passed for each invocation of the EXEC script. The default value is 100.

If the number of files exceeds the value specified for *MaxFiles*, **mmapplypolicy** invokes the external program multiple times.

mmapplypolicy

For more information about file list records, refer to the *IBM Spectrum Scale: Advanced Administration Guide*.

-D *yyyy-mm-dd[@hh:mm[:ss]]*

Specifies a date and optionally a (UTC) time as *year-month-day* at *hour:minute:second*.

The **mmapplypolicy** command evaluates policy rules as if it were running on the date and time specified by the **-D** flag. This can be useful for planning or testing policies, to see how the **mmapplypolicy** command would act in the future. If this flag is omitted, the **mmapplypolicy** command uses the current date and (UTC) time. If a date is specified but not a time, the time is assumed to be 00:00:00.

-e Causes **mmapplypolicy** to re-evaluate and revalidate the following conditions immediately before executing the policy action for each chosen file:

- That the `PATH_NAME` still leads to the chosen file, and that the `INODE` and `GENERATION` values are the same.
- That the rule (iRule) still applies to, and is a first matching rule for, the chosen file.

Note: The **-e** option is particularly useful with **-r**, but can be used apart from it. It is useful because in the time that elapses after the policy evaluation and up to the policy execution, it is possible that the chosen pathname no longer refers to the same inode (for example the original file was removed or renamed), or that some of the attributes of the chosen file have changed in some way so that the chosen file no longer satisfies the conditions of the rule. In general, the longer the elapsed time, the more likely it is that conditions have changed (depending on how the file system is being used). For example, if files are only written once and never renamed or erased, except by policy rules that call for deletion after an expiration interval, then it is probably not necessary to re-evaluate with the **-e** option.

For more information about **-r**, see *IBM Spectrum Scale: Advanced Administration Guide*.

-f *FileListPrefix*

Specifies the location (a path name or file name prefix or directory) in which the file lists for external pool and list operations are stored when either the **-I defer** or **-I prepare** option is chosen. The default location is *LocalWorkDirectory/mmapplypolicy.processid*.

-g *GlobalWorkDirectory*

Specifies a *global* directory to be used for temporary storage during **mmapplypolicy** command processing. The specified directory must exist within a shared file system. It must also be mounted and available for writing and reading from each of the nodes specified by the **-N** option. When both **-N** and **-g** are specified, **mmapplypolicy** uses high performance and fault-tolerant protocols during execution.

Note: The **-g** option should specify a directory (for temporary or work files) within a GPFS file system that is accessible from each node specified with the **-N** option. The directory can either be in the file system being operated upon by **mmapplypolicy** or in another file system.

There is no default value for **-g**.

If the **-g** option is specified, but not the **-s** option, the directory specified by **-g** is used for all temporary files required by **mmapplypolicy**. If both the **-g** and **-s** options are specified, temporary files may be stored in each. In general, temporary files that are only written and read by a single node are stored in the local work directory specified by the **-s** option, while temporary files that must be accessed by more than one node are stored in the global work directory specified by the **-g** option.

-I {yes | **defer** | **test** | **prepare**}

Specifies what actions the **mmapplypolicy** command performs on files:

yes

Indicates that all applicable policy rules are run, and the data movement between pools is done during the processing of the **mmapplypolicy** command. All defined external lists will be executed. This is the default action.

defer

Indicates that all applicable policy rules are run, but actual data movement between pools is deferred until the next **mmrestripefs** or **mmrestripefile** command. See also **-f FileListPrefix**.

test

Indicates that all policy rules are evaluated, but the **mmapplypolicy** command only displays the actions that would be performed had **-I defer** or **-I yes** been specified. There is no actual deletion of files or data movement between pools. This option is intended for testing the effects of particular policy rules.

prepare

Indicates that all policy execution is deferred and that **mmapplypolicy** only prepares file lists that are suitable for execution with the **-r** option. Records are written for each of the chosen files and are stored in one or more file lists, under a path name that is specified by the **-f** option or in the default local work directory. The actual data movement occurs when the command is rerun with the **-r** option.

-i InputFileList

Specifies the path name for a user-provided input file list. This file list enables you to specify multiple starter directories or files. It can be in either of the following formats:

simple format file list

A list of records with the following format:

PATH_NAME

Each record represents either a single file or a directory. When a directory is specified, the command processes the entire subtree that is rooted at the specified path name

File names can contain spaces and special characters; however, the special characters '\ ' and '\n' must be escaped with the '\ ' character similarly to the way **mmapplypolicy** writes path names in file lists for external pool and list operations.

The end-of-record character must be \n.

Example:

```
/mak/ea
/mak/old news
/mak/special\stuff
```

/usr/lpp/mmfs/samples/ilm/mmglobexpf.sample is an example of a script that can be used to generate simple format file lists.

expert format file list

A list of records with the following format:

INODE:GENERATION:path-length!PATH_NAME end-of-record-character

Each record represents exactly one file.

The INODE and GENERATION values must be specified in hexadecimal format (%11x). If you do not know the generation number or inode number, specify 0 and GPFS will look it up for you.

The *path-length* value must be specified in decimal format (%d). The *path-length* value is followed by the delimiter !.

The end-of-record character must be \n or \0.

mmapplypolicy

Example (the end-of-record characters are invisible):

```
00009a00:0:8!d14/f681
00009a01:1002:8!d14/f682
```

When you use an expert format file list, the directory scan phase is skipped and only the files that are specified with the *InputFileList* parameter are tested against the policy rules.

For more information, see the *IBM Spectrum Scale: Advanced Administration Guide*.

With either format, if a path name is not fully qualified, it is assumed to be relative to one of the following:

- the *Directory* parameter on the **mmapplypolicy** command invocation

Or

- the mount point of the GPFS file system, if *Device* is specified as the first argument

-L *n*

Controls the level of information displayed by the **mmapplypolicy** command. Larger values indicate the display of more detailed information. These terms are used:

candidate file

A file that matches a **MIGRATE**, **DELETE**, or **LIST** policy rule.

chosen file

A candidate file that has been scheduled for action.

These are the valid values for *n*:

- 0** Displays only serious errors.
- 1** Displays some information as the command runs, but not for each file. This is the default.
- 2** Displays each chosen file and the scheduled migration or deletion action.
- 3** Displays the same information as 2, plus each candidate file and the applicable rule.
- 4** Displays the same information as 3, plus each explicitly **EXCLUDE**ed or **LIST**ed file, and the applicable rule.
- 5** Displays the same information as 4, plus the attributes of candidate and **EXCLUDE**ed or **LIST**ed files.
- 6** Displays the same information as 5, plus non-candidate files and their attributes.

For examples and more information on this flag, see the section: *The mmapplypolicy -L command* in the *IBM Spectrum Scale: Problem Determination Guide*.

-M *name=value...*

Indicates a string substitution that will be made in the text of the policy rules before the rules are interpreted. This allows the administrator to reuse a single policy rule file for incremental backups without editing the file for each backup.

-m *ThreadLevel*

The number of threads that are created and dispatched within each **mmapplypolicy** process during the policy execution phase. The default value is 24.

-N {**a11** | **mount** | *Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the list of nodes that will run parallel instances of policy code in the GPFS home cluster. This command supports all defined node classes. The default is to run on the node where the **mmapplypolicy** command is running or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-n *DirThreadLevel...*

The number of threads that will be created and dispatched within each **mmapplypolicy** process during the directory scan phase. The default is 24.

-P *PolicyFile*

Specifies the name of the file containing the policy rules to be applied. If not specified, the policy rules currently in effect for the file system are used. Use the **mmlspolicy** command to display the current policy rules.

-q When specified, **mmapplypolicy** dispatches bunches of files from the file lists specified by the **-r** option in a round-robin fashion, so that the multithreaded (**-m**) and node parallel (**-N**) policy execution works on all the file lists "at the same time." When **-q** is not specified, policy execution works on the file lists sequentially. In either case bunches of files are dispatched for parallel execution to multiple threads (**-m**) on each of the possibly multiple nodes (**-N**).

-r *FileListPathname...*

Specifies one or more file lists of files for policy execution. The file lists that were used as input for **-r** were created by issuing **mmapplypolicy** with the **-I prepare** flag. You can specify several file lists by doing one of the following:

- Provide the path name of a directory of file lists, **or**
- Specify the **-r** option several times, each time with the path name of a different file list.

You can use this parameter to logically continue where **mmapplypolicy** left off when you specified the **-I prepare** option. To do this, invoke **mmapplypolicy** with all the same parameters and options (except the **-I prepare** option), and now substitute the **-r** option for **-f**. In between the invocations, you can process, reorder, filter, or edit the file lists that were created when you invoked **-I prepare**. You can specify any or all of the resulting file lists with the **-r** option.

The format of the records in each file list file can be expressed as:

```
iAggregate:WEIGHT:INODE:GENERATION:SIZE:iRule:resourceId:attr_flags:
path-length|!PATH_NAME:pool-length!POOL_NAME
[;show-length>!SHOW]end-of-record-character
```

For more information about file list records, refer to the *IBM Spectrum Scale: Advanced Administration Guide*.

-S *SnapshotName*

Specifies the name of a global snapshot for file system backup operations. The name appears as a subdirectory of the **.snapshots** directory in the file system root and can be found with the **mmlssnapshot** command.

Note: GPFS snapshots are read-only. Do not use migration or deletion rules with **-S SnapshotName**.

-s *LocalWorkDirectory*

Specifies the directory to be used for temporary storage during **mmapplypolicy** command processing.

The default directory is **/tmp**. The **mmapplypolicy** command stores lists of candidate and chosen files in temporary files within this directory.

When you execute **mmapplypolicy**, it creates several temporary files and file lists. If the specified file system or directories contain many files, this can require a significant amount of temporary storage. The required storage is proportional to the number of files (NF) being acted on and the average length of the path name to each file (AVPL). To make a rough estimate of the space required, estimate NF and assume an AVPL of 80 bytes. With an AVPL of 80, the space required is roughly (300 X NF) bytes of temporary space.

--choice-algorithm {**best** | **exact** | **fast**}

Specifies one of the following types of algorithms that the policy engine is to use when selecting candidate files:

best

Chooses the optimal method based on the rest of the input parameters.

mmapplypolicy

exact

Sorts all of the candidate files completely by weight, then serially considers each file from highest weight to lowest weight, choosing feasible candidates for migration, deletion, or listing according to any applicable rule **LIMITs** and current storage-pool occupancy. This is the default.

fast

Works together with the parallelized **-g /shared-tmp -N** node-list selection method. The **fast** choice method does not completely sort the candidates by weight. It uses a combination of statistical, heuristic, and parallel computing methods to favor higher weight candidate files over those of lower weight, but the set of chosen candidates may be somewhat different than those of the **exact** method, and the order in which the candidates are migrated, deleted, or listed is somewhat more random. The **fast** method uses statistics gathered during the policy evaluation phase. The **fast** choice method is especially fast when the collected statistics indicate that either all or none of the candidates are feasible.

--max-merge-files

Specifies the maximum number of files to be passed as input to the sort/merge command.

--max-sort-bytes

Specifies the maximum number of bytes to be passed as one or more input files to the sort command. This does not apply to merges, when each of the input files has already been sorted.

--scope {filesystem | inodespace | fileset}

If *Device* is specified, the directory traversal starts at the root of the file system. **--scope** indicates one of the following levels of scope to be applied to the policy scan:

filesystem

The scan will involve the objects in the entire file system subtree pointed to by the *Directory* parameter. This is the default.

inodespace

The scope is limited to objects in the same single inode space from which the directory pointed to by the *Directory* parameter is allocated. The scan may span more than one fileset, if those filesets share the same inode space.

fileset

The scope of the scan is limited to the objects in the same fileset as the directory pointed to by the *Directory* parameter.

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

--other-sort-options SortOptions

Passes options to the sort command (either the default sort command provided with the operating system, or an alternative sort command specified by the **--sort-command** parameter).

--single-instance

Ensures that, for the specified file system, only one instance of **mmapplypolicy** invoked with the **--single-instance** option can execute at one time. If another instance of **mmapplypolicy** invoked with the **--single-instance** option is currently executing, this invocation will do nothing but terminate.

--sort-buffer-size *Size*

Limits the memory usage of any sort subprocesses executed by **mmapplypolicy**. The *Size* value is passed to the sort command on operating systems that support the **--buffer-size** option and can be specified in any syntax that is accepted by **--buffer-size** (for example, 20% or 1M).

--sort-command *SortCommand*

Specifies the fully-qualified path name for a Posix-compliant sort command to be used instead of the default, standard command provided by the operating system.

Before specifying an alternative sort command (and for information about a suggested sort command), see the topic about improving the performance of **mmapplypolicy** with the **--sort-command** parameter in *IBM Spectrum Scale: Advanced Administration Guide*.

--split-filelists-by-weight

Specifies that each of the generated file lists contain elements with the same **WEIGHT** value. This can be useful in conjunction with the **LIST** rule and the **WEIGHT (DIRECTORY_HASH)** clause. In this case, each generated list will contain files from the same directory.

Note: If you want all of the files from a given directory to appear in just one list, you might have to specify a sufficiently large **-B** value.

--split-margin *n.n*

A floating-point number that specifies the percentage within which the **fast-choice** algorithm is allowed to deviate from the **LIMIT** and **THRESHOLD** targets specified by the policy rules. For example if you specified a **THRESHOLD** number of 80% and a split-margin value of 0.2, the **fast-choice** algorithm could finish choosing files when it reached 80.2%, or it might choose files that bring the occupancy down to 79.8%. A nonzero value for split-margin can greatly accelerate the execution of the **fast-choice** algorithm when there are many small files. The default is 0.2.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmapplypolicy** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This command displays the actions that would occur if a policy were applied, but does not apply the policy at this time:

```
mmapplypolicy fsl -P policyfile -I test
```

The system displays output similar to:

```
[I] GPFS current data pool utilization in KB and %
sp1   9728   19531264   0.049807%
sp2   4608   19531264   0.023593%
```

mmapplypolicy

```
system 105216 19531264 0.538706%
[I] Loaded policy rules from fs1.pol.
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP = 2009-02-27@20:00:22 UTC
parsed 2 Placement Rules, 0 Restore Rules, 3 Migrate/Delete/Exclude Rules,
  0 List Rules, 0 External Pool/List Rules
RULE 'sp1' SET POOL 'sp1' WHERE name like '%.sp1' or name like '%.tmp'
RULE 'default' SET POOL 'system'

RULE 'exclude *.save files' EXCLUDE WHERE NAME LIKE '%.save'

/* Deletion rule */
RULE 'delete' DELETE FROM POOL 'sp1' WHERE NAME LIKE '%.tmp'

/* Migration rule */
RULE 'migration to system pool' MIGRATE FROM POOL 'sp1' TO POOL 'system' WHERE NAME LIKE '%sp1%'
[I] Directories scan: 11 files, 1 directories, 0 other objects, 0 'skipped' files and/or errors.
[I] Inodes scan: 11 files, 1 directories, 0 other objects, 0 'skipped' files and/or errors.
[I] Summary of Rule Applicability and File Choices:
Rule# Hit_Cnt KB_Hit Chosen KB_Chosen KB_Ill Rule
  0     3     1536    0     0     0     RULE 'exclude *.save files' EXCLUDE WHERE(.)
  1     3     1536    3     1536    0     RULE 'delete' DELETE FROM POOL 'sp1' WHERE(.)
  2     2     1024    2     1024    0     RULE 'migration to system pool' MIGRATE FROM POOL \
    'sp1' TO POOL 'system' WHERE(.)

[I] Files with no applicable rules: 4.

[I] GPFS Policy Decisions and File Choice Totals:
Chose to migrate 1024KB: 2 of 2 candidates;
Chose to premigrate 0KB: 0 candidates;
Already co-managed 0KB: 0 candidates;
Chose to delete 1536KB: 3 of 3 candidates;
Chose to list 0KB: 0 of 0 candidates;
0KB of chosen data is illplaced or illreplicated;
Predicted Data Pool Utilization in KB and %:
sp1     7168  19531264  0.036700%
sp2     4608  19531264  0.023593%
system 106240 19531264  0.543948%
```

2. This command applies a policy immediately:

```
mmapplypolicy fs1 -P policyfile
```

The system displays output similar to:

```
[I] GPFS current data pool utilization in KB and %
sp1     9728  19531264  0.049807%
sp2     4608  19531264  0.023593%
system 105216 19531264  0.538706%
[I] Loaded policy rules from fs1.pol.
Evaluating MIGRATE/DELETE/EXCLUDE rules with CURRENT_TIMESTAMP = 2009-02-27@20:2
5:34 UTC
parsed 2 Placement Rules, 0 Restore Rules, 3 Migrate/Delete/Exclude Rules,
  0 List Rules, 0 External Pool/List Rules
RULE 'sp1' SET POOL 'sp1' WHERE name like '%.sp1' or name like '%.tmp'
RULE 'default' SET POOL 'system'

RULE 'exclude *.save files' EXCLUDE WHERE NAME LIKE '%.save'

/* Deletion rule */
RULE 'delete' DELETE FROM POOL 'sp1' WHERE NAME LIKE '%.tmp'

/* Migration rule */
RULE 'migration to system pool' MIGRATE FROM POOL 'sp1' TO POOL 'system' WHERE NAME LIKE '%sp1%'
[I] Directories scan: 11 files, 1 directories, 0 other objects, 0 'skipped' files and/or errors.
[I] Inodes scan: 11 files, 1 directories, 0 other objects, 0 'skipped' files and/or errors.
[I] Summary of Rule Applicability and File Choices:
Rule# Hit_Cnt KB_Hit Chosen KB_Chosen KB_Ill Rule
  0     3     3072    0     0     0     RULE 'exclude *.save files' EXCLUDE WHERE(.)
```



```

1      3      3072    3      3072    0      RULE 'delete' DELETE FROM POOL 'sp1' WHERE(.)
2      2      2048    2      2048    0      RULE 'migration to system pool'MIGRATE FROM POOL \
                                     'sp1' TO POOL 'system' WHERE(.)

```

[I] Files with no applicable rules: 4.

[I] GPFS Policy Decisions and File Choice Totals:

```

Chose to migrate 2048KB: 2 of 2 candidates;
Chose to premigrate 0KB: 0 candidates;
Already co-managed 0KB: 0 candidates;
Chose to delete 3072KB: 3 of 3 candidates;
Chose to list 0KB: 0 of 0 candidates;
0KB of chosen data is illplaced or illreplicated;
Predicted Data Pool Utilization in KB and %:
sp1      4608    19531264    0.023593%
sp2      4608    19531264    0.023593%
system  107264   19531264    0.549191%

```

```

[I] A total of 5 files have been migrated, deleted or processed by an EXTERNAL E
XEC/script;
    0 'skipped' files and/or errors.

```

Additional examples of GPFS policies and using the **mmapplypolicy** command are in the *IBM Spectrum Scale: Advanced Administration Guide*.

See also

- “mmchpolicy command” on page 391
- “mmcrsnapshot command” on page 431
- “mmlspolicy command” on page 552
- “mmlssnapshot command” on page 563
- “mmsnapdir command” on page 674

Location

/usr/lpp/mmfs/bin

mmauth

mmauth command

Manages secure access to GPFS file systems.

Synopsis

```
mmauth genkey {new | commit | propagate [-N {Node[,Node...]} | NodeFile | NodeClass]}
```

or

```
mmauth add RemoteClusterName -k KeyFile [-1 CipherList]
```

or

```
mmauth update RemoteClusterName {[-C NewClusterName]} [-k KeyFile] [-1 CipherList]}
```

or

```
mmauth delete {RemoteClusterName | all}
```

or

```
mmauth grant {RemoteClusterName | all} -f {Device | all} [-a {rw | ro}] [-r {uid:gid | no}]
```

or

```
mmauth deny {RemoteClusterName | all} -f {Device | all}
```

or

```
mmauth show [RemoteClusterName | all | ciphers]
```

or

```
mmauth gencert --cname CanonicalName --cert ServerCertificateFile --out OutputKeystoreFile  
              --label ClientCertificateLabel [--pwd-file KeystorePasswordFile]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmauth** command prepares a cluster to grant secure access to file systems owned locally. The **mmauth** command also prepares a cluster to receive secure access to file systems owned by another cluster. Use the **mmauth** command to generate a public/private key pair for the local cluster. A public/private key pair must be generated on both the cluster owning the file system and the cluster desiring access to the file system. The administrators of the clusters are responsible for exchanging the public portion of the public/private key pair. Use the **mmauth** command to add or delete permission for a cluster to mount file systems owned by the local cluster.

When a cluster generates a new public/private key pair, administrators of clusters participating in remote file system mounts are responsible for exchanging their respective public key file `/var/mmfs/ssl/id_rsa.pub` generated by this command.

The administrator of a cluster desiring to mount a file system from another cluster must provide the received key file as input to the **mmremotecenter** command. The administrator of a cluster allowing another cluster to mount a file system must provide the received key file to the **mmauth** command.

The keyword appearing after **mmauth** determines which action is performed:

add

Adds a cluster and its associated public key to the list of clusters authorized to connect to this cluster for the purpose of mounting file systems owned by this cluster.

delete

Deletes a cluster and its associated public key from the list of clusters authorized to mount file systems owned by this cluster.

deny

Denies a cluster the authority to mount a specific file system owned by this cluster.

gencert

Creates a client keystore with the keys and certificates required to communicate with the ISKLM key server.

genkey

Controls the generation and propagation of the OpenSSL key files:

new

Generates a new public/private key pair for this cluster. The key pair is placed in `/var/mmfs/ssl`. This must be done at least once before `cipherList`, the GPFS configuration parameter that enables GPFS with OpenSSL, is set.

The new key is in addition to the currently in effect committed key. Both keys are accepted until the administrator runs `mmauth genkey commit`.

commit

Commits the new public/private key pair for this cluster. Once `mmauth genkey commit` is run, the old key pair will no longer be accepted, and remote clusters that have not updated their keys (by running `mmauth update` or `mmremotecenter update`) will be disconnected.

propagate

Ensures that the currently in effect key files are placed in `/var/mmfs/ssl` on the nodes specified with the `-N` parameter. This may be necessary if the key files are lost and `adminMode central` is in effect for the cluster.

grant

Allows a cluster to mount a specific file system owned by this cluster.

show

Shows the list of clusters authorized to mount file system owned by this cluster.

update

Updates the public key and other information associated with a cluster authorized to mount file systems owned by this cluster.

When the local cluster name (or `.`) is specified, `mmauth update -l` can be used to set the `cipherList` value for the local cluster. Note that you cannot use this command to change the name of the local cluster. Use the `mmchcluster` command for this purpose.

Parameters

`-N {Node[,Node...] | NodeFile | NodeClass}`

Specifies the nodes on which the key files should be restored. The default is `-N all`.

For general information on how to specify node names, see the following *IBM Spectrum Scale: Administration and Programming Reference* topic: "Specifying nodes as input to GPFS commands" on page 3.

This command does not support a `NodeClass` of `mount`.

RemoteClusterName

Specifies the remote cluster name requesting access to local GPFS file systems.

mmauth

all

Indicates all remote clusters defined to the local cluster.

ciphers

Shows the supported ciphers.

Options

-a {**rw** | **ro**}

Specifies the type of access allowed:

ro Specifies read-only access.

rw Specifies read/write access. This is the default.

-C *NewClusterName*

Specifies a new, fully-qualified cluster name for the already-defined cluster *RemoteClusterName*.

-f {*Device* | **all**}

Specifies the device name for a file system owned by this cluster. The *Device* argument is required. If **all** is specified, the command applies to all file systems owned by this cluster at the time that the command is issued.

-k *KeyFile*

Specifies the public key file generated by the **mmauth** command in the cluster requesting to remotely mount the local GPFS file system.

-l *CipherList*

Sets the security mode for communications between the current cluster and the remote cluster that is specified in the *RemoteClusterName* parameter. There are three security modes:

EMPTY

The sending node and the receiving node do not authenticate each other, do not encrypt transmitted data, and do not check data integrity.

AUTHONLY

The sending and receiving nodes authenticate each other, but they do not encrypt transmitted data and do not check data integrity. This mode is the default in IBM Spectrum Scale V4.2 or later.

Cipher

The sending and receiving nodes authenticate each other, encrypt transmitted data, and check data integrity. To set this mode, you must specify the name of a supported cipher, such as AES128-GCM-SHA256.

For more information about the security mode and supported ciphers, see the topic *Security mode* in the *IBM Spectrum Scale: Advanced Administration Guide*.

-r {*uid:gid* | **no**}

Specifies a root credentials remapping (*root squash*) option. The UID and GID of all processes with root credentials from the remote cluster will be remapped to the specified values. The default is not to remap the root UID and GID. The *uid* and *gid* must be specified as unsigned integers or as symbolic names that can be resolved by the operating system to a valid UID and GID. Specifying **no**, **off**, or **DEFAULT** turns off the remapping.

For more information, see the *IBM Spectrum Scale: Advanced Administration Guide* and search on *root squash*.

--cname *CanonicalName*

Specifies the canonical name of the client used in the certificate.

--cert *ServerCertificateFile*

Specifies the path name to a file containing an ISKLM certificate.

--out *OutputKeystoreFile*

Specifies the path name for the file that will contain the keystore.

--pwd-file *KeystorePasswordFile*

Specifies the keystore password file. If omitted, you will be prompted to enter the keystore password. A maximum of 20 characters are allowed. The **--pwd** *KeystorePassword* option is considered deprecated and may be removed in a future release.

--label *ClientCertificateLabel*

Specifies the label of the client certificate within the keystore. A maximum of 20 characters are allowed.

Exit status

0 Successful completion. After a successful completion of the **mmauth** command, the configuration change request will have been propagated to all nodes in the cluster.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmauth** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This is an example of an **mmauth genkey new** command:

```
mmauth genkey new
```

The output is similar to this:

```
Generating RSA private key, 512 bit long modulus
.....+++++
e is 65537 (0x10001)
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

2. This is an example of an **mmauth genkey commit** command:

```
mmauth genkey commit
```

The output is similar to this:

```
mmauth: Command successfully completed
mmauth: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

3. This is an example of an **mmauth add** command:

```
mmauth add clustA.kgn.ibm.com -k /u/admin/keys/clustA.pub
```

The output is similar to this:

```
mmauth: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

4. This is an example of an **mmauth update** command:

```
mmauth update clustA.kgn.ibm.com -k /u/admin/keys/clustA_new.pub
```

The output is similar to this:

mmauth

mmauth: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

5. This is an example of an **mmauth grant** command:

```
mmauth grant clustA.kgn.ibm.com -f /dev/gpfs1 -a ro
```

The output is similar to this:

mmauth: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

6. This is an example of an **mmauth show** command:

```
mmauth show all
```

The output is similar to this:

```
Cluster name:      clustA.kgn.ibm.com
Cipher list:       AES128-SHA
SHA digest:        a3917c8282fca7a27d951566940768dcd241902b
File system access: gpfs1 (ro)

Cluster name:      clustB.kgn.ibm.com (this cluster)
Cipher list:       AES128-SHA
SHA digest:        6ba5e3c1038246fe30f3fc8c1181fbb2130d7a8a
SHA digest (new):  3c1038246fe30f3fc8c1181fbb2130d7a8a9ab4d
File system access: (all rw)
```

For **clustB.kgn.ibm.com**, the **mmauth genkey new** command has been issued, but the **mmauth genkey commit** command has not yet been issued.

For more information on the SHA digest, see the *IBM Spectrum Scale: Problem Determination Guide* and search on *SHA digest*.

See also

- “mmremotefs command” on page 624
- “mmremotecluster command” on page 621

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: Advanced Administration Guide*.

Location

```
/usr/lpp/mmfs/bin
```

mmbackup command

Performs a backup of a GPFS file system or independent fileset to a Tivoli Storage Manager (TSM) server.

Synopsis

```
mmbackup {Device | Directory} [-t {full | incremental}]
        [-N {Node[,Node...] | NodeFile | NodeClass}]
        [-g GlobalWorkDirectory] [-s LocalWorkDirectory]
        [-S SnapshotName] [-f] [-q] [-v] [-d]
        [-a IscanThreads] [-n DirThreadLevel]
        [-m ExecThreads | [[--expire-threads ExpireThreads] [--backup-threads BackupThreads]]]
        [-B MaxFiles | [[--max-backup-count MaxBackupCount] [--max-expire-count MaxExpireCount]]]
        [--max-backup-size MaxBackupSize] [--qos QosClass] [--quote | --noquote]
        [--rebuild] [--scope {filesystem | inodespace}]
        [--tsm-servers TSMServer[,TSMServer...]]
        [--tsm-errorlog TSMErrorLogFile] [-L n] [-P PolicyFile]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

Use the **mmbackup** command to back up the user data from a GPFS file system or independent fileset to a TSM server or servers. The **mmbackup** command can only be used to back up file systems owned by the local cluster.

Attention: In GPFS V4.1 and later, a full backup (-t full) with **mmbackup** is required if a full backup has never been performed with GPFS 3.3 or later. For more information, see the topic *File systems backed up using GPFS 3.2 or earlier versions of mmbackup* in the *IBM Spectrum Scale: Advanced Administration Guide*.

The TSM Backup-Archive client must be installed and at the same version on all the nodes that will be executing the **mmbackup** command or named in a node specification with -N. For more information about TSM requirements for the **mmbackup** command, see the topic *GPFS port usage* in the *IBM Spectrum Scale: Advanced Administration Guide*.

You can run multiple instances of **mmbackup**, as long as they are on different file systems.

If you are planning to use Tivoli Storage Manager to back up IBM Spectrum Scale file systems, see the topic *Backup considerations for using Tivoli Storage Manager* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the topic *Configuration reference for using Tivoli Storage Manager with IBM Spectrum Scale* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

Device

The device name for the file system to be backed up. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

Directory

Specifies either the mount point of a GPFS file system or the path to an independent fileset root to back up. **/gpfs/fs0** can be used to specify the GPFS file system called **fs0**.

Note: A snapshot directory path is not permitted. To back up a snapshot, use -S *SnapshotName* instead. Do not use a subdirectory path as this will lead to inconsistent backups.

mmbackup

-t {full | incremental}

Specifies whether to perform a full backup of all of the files in the file system, or an incremental backup of only those files that have changed since the last backup was performed. The default is an **incremental** backup.

A full backup will expire all GPFS 3.2 format TSM inventory if all previous backups have been incremental and the GPFS 3.2 backup format had been in use previously. If **mmbackup** on GPFS 3.2 was first used, and then only incremental backups were done on GPFS 3.4 or 3.5, then TSM will still contain 3.2 format backup inventory. This inventory will automatically be marked for expiration by **mmbackup** after a successful full or incremental backup.

Note: Do not use **-t full** with an unlinked fileset. Use an **EXCLUDE** statement to exclude directories or files from the backup operation.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the list of nodes that will run parallel instances of the backup process. The TSM Backup-Archive client must be installed on all nodes specified with this parameter. This command supports all defined node classes. The default is to run only on the node where the **mmbackup** command is running or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-g GlobalWorkDirectory

Specifies the directory to be used for temporary files that need to be shared between the **mmbackup** worker nodes. Defaults to the value specified with the **-s** option or **/tmp**.

-s LocalWorkDirectory

Specifies the local directory to be used for temporary storage during **mmbackup** command processing. The default directory is **/tmp**. A *LocalWorkDirectory* must exist on each node used to run the **mmbackup** command or specified in a node specification with **-N**.

-S SnapshotName

Specifies the name of a global snapshot for any backup operations or a fileset-level snapshot if **--scope inodespace** is also specified for a fileset backup. The snapshot must be created before the **mmbackup** command is used. Snapshot names can be found with the **mmlsnapshot** command. If a fileset is not present in the named snapshot and fileset-level backup is invoked with **--scope inodespace**, an error is returned. If a fileset-level snapshot name is used with a file system backup, an error is returned.

The use of **-S SnapshotName** is recommended because it provides **mmbackup** and TSM a consistent view of GPFS from which to perform backup. Deletion of the snapshot used for backup can be performed using the **mmdelsnapshot** command after **mmbackup** completes.

-f Specifies that processing should continue when unlinked filesets are detected. All files that belong to unlinked filesets will be ignored.

Note: Because **-f** has a large impact on performance, avoid using it unless performing a backup operation with unlinked filesets is absolutely necessary.

-q Performs a query operation prior to beginning **mmbackup**. The TSM server may have data stored already that is not recognized as having been backed up by **mmbackup** and its own shadow database. To properly compute the set of files that currently need to be backed up, **mmbackup** can perform a TSM query and process the results to update its shadow database. Use the **-q** switch to perform this query and then immediately commence the requested backup operation.

Note: Do not use **-q** with the **--rebuild** parameter.

-v Specifies verbose message output. Use this flag to cause **mmbackup** to issue more verbose messages about its processing. See also "Environment" on page 285.

- d Gathers debugging information that is useful to the IBM Support Center when diagnosing problems.
- a *IscanThreads*
Specifies the number of threads and sort pipelines each node will run during parallel inode scan and policy evaluation. The default value is 2.
- n *DirThreadLevel*
Specifies the number of threads that will be created and dispatched within each **mmapplypolicy** process during the directory scan phase. The default value is 24.
- m *ExecThreads*
Specifies the number of threads created and dispatched within each **mmapplypolicy** process during the policy execution phase. The default value for **mmapplypolicy** is 24; however, the default value for **mmbackup** is 1. This option cannot be used in conjunction with the **--expire-threads** and **--backup-threads** options.
- expire-threads** *ExpireThreads*
Specifies the number of worker threads permitted on each node to perform **dsmc expire** tasks in parallel. This option cannot be used in conjunction with the **-m** option. Valid values are 1 - 32. The default value is 4.
- backup-threads** *BackupThreads*
Specifies the number of worker threads permitted on each node to perform **dsmc selective** or **dsmc incremental** tasks in parallel. This option cannot be used in conjunction with the **-m** option. Valid values are 1 - 32. The default value is 1.
- B *MaxFiles*
Specifies the maximum number of objects in a bunch for each invocation of the **mmapplypolicy** EXEC script. If this option is not specified, the ideal bunch count will be automatically computed. Valid values are 100 - 8192.
- max-backup-count** *MaxBackupCount*
Specifies the maximum number of objects in a bunch for each **dsmc selective** or **dsmc incremental** command. This option cannot be used in conjunction with the **-B** option. Valid values are 100 - 8192.
- max-expire-count** *MaxExpireCount*
Specifies the maximum number of objects in a bunch for each **dsmc expire** command. This option cannot be used in conjunction with the **-B** option. Valid values are 100 - 8192.
- max-backup-size** *MaxBackupSize*
Specifies a policy limit on the content size in kilobytes for each **dsmc selective** or **dsmc incremental** command.
- qos** *QoSClass*
Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:
 - maintenance** This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.
 - other** This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

mmbackup

--quote | --noquote

Specifies whether to decorate (or not to decorate) file-list entries with quotation marks. The **mmbackup** command uses file lists to convey the lists of files to the TSM Backup-Archive client program. The file lists may or may not require each file name to be surrounded by quotation marks depending on TSM client configuration options. If certain TSM client configuration options are in use, the quotation marks should not be added to file-list entries. Use the **--noquote** option in these instances.

--rebuild

Specifies whether to rebuild the **mmbackup** shadow database from the inventory of the TSM server. This option is similar to the **-q** option; however, no backup operation proceeds after the shadow database is rebuilt. This option should be used if the shadow database of **mmbackup** is known to be out of date, but the rebuilding operation must be done at a time when the TSM server is less loaded than during the normal time **mmbackup** is run.

If there are backup files with the old snapshot name */Device/.snapshots/.mmbuSnapshot* in the inventory of the TSM server, those files will be expired from the TSM server after the shadow database is rebuilt and any successful incremental or full backup completes.

Note: Do not use **--rebuild** with the **-q** parameter.

--scope {filesystem | inodespace}

Specifies that one of the following traversal scopes be applied to the policy scan and backup candidate selection:

filesystem

Scans all the objects in the file system specified by *Device* or mounted at the *Directory* specified. This is the default behavior.

inodespace

Specifies that the scan will be limited in scope to objects in the same single inode space from which the *Directory* is allocated. The scan might span more than one fileset if those filesets share the same inode space; for example, dependent filesets.

--tsm-servers *TSMServer* [, *TSMServer*...]

Specifies the name of the TSM server or servers to be used for this backup. The TSM servers specified will each be used for the backup task specified.

If this option is not specified, the **mmbackup** command will backup to the servers that are specified in the **dsm.sys** file.

--tsm-errorlog *TSMErrorLogFile*

Specifies the pathname of the log file to pass to TSM Backup-Archive client commands

-L *n*

Controls the level of information displayed by the **mmbackup** command. Larger values indicate the display of more detailed information. *n* should be one of the following values:

- 0 Displays only serious errors. This is the default.
- 1 Displays some information as the command runs, but not for each file.
- 2 Displays each chosen file and the scheduled migration or deletion action.
- 3 Displays the same information as 2, plus each candidate file and the applicable rule.
- 4 Displays the same information as 3, plus each explicitly **EXCLUDEEed** or **LISTed** file, and the applicable rule.
- 5 Displays the same information as 4, plus the attributes of candidate and **EXCLUDEEed** or **LISTed** files.
- 6 Displays the same information as 5, plus non-candidate files and their attributes.

-P PolicyFile

Specifies a customized policy rules file for the backup.

Environment

The behavior of **mmbackup** can be influenced by several environment variables when set.

Variables that apply to TSM Backup-Archive client program dsmsc**MMBACKUP_DSMS_MISC**

The value of this variable is passed as arguments to **dsmsc restore** and **dsmsc query {backup,incl excl,session}** commands.

MMBACKUP_DSMS_BACKUP

The value of this variable is passed as arguments to **dsmsc**, **dsmsc selective**, and **dsmsc incremental** commands.

MMBACKUP_DSMS_EXPIRE

The value of this variable is passed as arguments to **dsmsc expire** commands.

Variables that change mmbackup output progress reporting**MMBACKUP_PROGRESS_CONTENT**

Controls what progress information is displayed to the user as **mmbackup** runs. It is a bit field with the following bit meanings:

0x01

Specifies that basic text progress for each server is to be displayed.

0x02

Specifies that additional text progress for phases within each server is to be displayed.

0x04

Specifies that numerical information about files being considered is to be displayed.

MMBACKUP_PROGRESS_INTERVAL

Controls how frequently status callouts are made. The value is the minimum number of seconds between calls to the **MMBACKUP_PROGRESS_CALLOUT** script or program. It does not affect how frequently messages are displayed, except for the messages of **MMBACKUP_PROGRESS_CONTENT** category 0x04.

MMBACKUP_PROGRESS_CALLOUT

Specifies the path to a program or script to be called with a formatted argument, as described in the topic *MMBACKUP_PROGRESS_CALLOUT environment variable* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Variables that change mmbackup debugging facilities

In case of a failure, certain debugging and data collection can be enabled by setting the specified environment variable value.

DEBUGmmbackup

This variable controls what debugging features are enabled. It is interpreted as a bitmask with the following bit meanings:

0x001

Specifies that basic debug messages are printed to STDOUT. There are multiple components that comprise **mmbackup**, so the debug message prefixes can vary. Some examples include:

```
mmbackup:mbackup.sh
DEBUGtsbackup33:
```

0x002

Specifies that temporary files are to be preserved for later analysis.

mmbackup

0x004

Specifies that all **dsmc** command output is to be mirrored to STDOUT.

DEBUGmmcmi

This variable controls debugging facilities in the **mmbackup** helper program **mmcmi**, which is used when the cluster **minReleaseLevel** is less than 3.5.0.11.

DEBUGtsbuhelper

This variable controls debugging facilities in the **mmbackup** helper program **tsbuhelper**, which is used when the cluster **minReleaseLevel** is greater than or equal to 3.5.0.11.

Variables that change mmbackup record locations

MMBACKUP_RECORD_ROOT

Specifies an alternate directory name for storing all temporary and permanent records for the backup. The directory name specified must be an existing directory and it cannot contain special characters (for example, a colon, semicolon, blank, tab, or comma).

The directory specified for **MMBACKUP_RECORD_ROOT** must be accessible from each node specified with the **-N** option.

Exit status

- 0 Successful completion. All of the eligible files were backed up.
- 1 Partially successful completion. Some files, but not all eligible files, were backed up. The shadow database or databases reflect the correct inventory of the TSM server. Invoke **mmbackup** again to complete the backup of eligible files.
- 2 A failure occurred that prevented backing up some or all files or recording any progress in the shadow database or databases. Correct any known problems and invoke **mmbackup** again to complete the backup of eligible files. If some files were backed up, using the **-q** or **--rebuild** option can help avoid backing up some files additional times.

Security

You must have root authority to run the **mmbackup** command.

The node on which the command is issued, as well as all other TSM Backup-Archive client nodes, must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To perform an incremental backup of the file system **gpfs0**, issue this command:

```
mmbackup gpfs0
```

The system displays information similar to:

```
-----  
mmbackup: Backup of /gpfs/gpfs0 begins at Mon Apr 7 15:37:50 EDT 2014.  
-----
```

```
Mon Apr 7 15:38:04 2014 mmbackup:Scanning file system gpfs0  
Mon Apr 7 15:38:14 2014 mmbackup:Determining file system changes for gpfs0 [balok1].  
Mon Apr 7 15:38:14 2014 mmbackup:changed=364, expired=0, unsupported=0 for server [balok1]  
Mon Apr 7 15:38:14 2014 mmbackup:Sending files to the TSM server [364 changed, 0 expired].  
mmbackup: TSM Summary Information:  
Total number of objects inspected:      364  
Total number of objects backed up:      364  
Total number of objects updated:        0  
Total number of objects rebound:       0  
Total number of objects deleted:        0  
Total number of objects expired:        0
```

```
Total number of objects failed:      0
Total number of bytes transferred:    2179695902
```

```
-----
mmbackup: Backup of /gpfs/gpfs0 completed successfully at Mon Apr 7 15:41:09 EDT 2014.
-----
```

- To recreate a shadow database for **gpfs0**, issue this command:

```
mmbackup gpfs0 --rebuild
```

The system displays information similar to:

```
-----
mmbackup: Shadow database rebuild of /gpfs/gpfs0 begins at Tue Apr 8 09:44:59 EDT 2014.
-----
```

```
Tue Apr  8 09:45:11 2014 mmbackup:Querying files currently backed up in TSM server:balok1.
Tue Apr  8 09:45:14 2014 mmbackup:Built query data file from TSM server: balok1 rc = 0
Tue Apr  8 09:45:17 2014 mmbackup:Scanning file system gpfs0
Tue Apr  8 09:45:26 2014 mmbackup:Reconstructing previous shadow file /gpfs/gpfs0/.mmbackupShadow.1.balok1 from
query data for balok1
Tue Apr  8 09:45:26 2014 mmbackup:Done with shadow file database rebuilds
```

```
-----
mmbackup: Shadow database rebuild of /gpfs/gpfs0 completed successfully at Tue Apr 8 09:45:26 EDT 2014.
-----
```

- To perform an incremental backup of the file system **gpfs0** with more progress information displayed, first issue this command:

```
export MMBACKUP_PROGRESS_CONTENT=3
```

Next, issue the **mmbackup** command:

```
mmbackup gpfs0
```

The system displays information similar to:

```
-----
mmbackup: Backup of /gpfs/gpfs0 begins at Mon Apr 7 16:02:28 EDT 2014.
-----
```

```
Mon Apr  7 16:02:33 2014 mmbackup:Begin checking server and shadow file for balok1
Mon Apr  7 16:02:37 2014 mmbackup:Querying TSM server balok1 for options
Mon Apr  7 16:02:40 2014 mmbackup:Found old shadow DB for balok1 present in /gpfs/gpfs0/.mmbackupShadow.1.balok1
Mon Apr  7 16:02:40 2014 mmbackup:Checking format version of old shadow DB
Mon Apr  7 16:02:40 2014 mmbackup:Found old shadow DB version: 1400
Mon Apr  7 16:02:40 2014 mmbackup:Previous shadow /gpfs/gpfs0/.mmbackupShadow.1.balok1 state: present
Mon Apr  7 16:02:40 2014 mmbackup:Generating policy rules file:/var/mmfs/mmbackup/.mmbackupRules.gpfs0 to
use /gpfs/gpfs0/.mmbackupCfg/BAexecScript.gpfs0
Mon Apr  7 16:02:42 2014 mmbackup:Completed policy rule generation.  2 Exclude Dir directives, 1 Exclude File
directives, 1 Include directives, 0 Warnings.
Mon Apr  7 16:02:42 2014 mmbackup:Scanning file system gpfs0
Mon Apr  7 16:02:51 2014 mmbackup:File system scan of gpfs0 is complete.
Mon Apr  7 16:02:51 2014 mmbackup:Calculating backup and expire lists for server balok1
Mon Apr  7 16:02:51 2014 mmbackup:Determining file system changes for gpfs0 [balok1].
Mon Apr  7 16:02:51 2014 mmbackup:changed=364, expired=0, unsupported=0 for server [balok1]
Mon Apr  7 16:02:51 2014 mmbackup:Finished calculating lists [364 changed, 0 expired] for server balok1.
Mon Apr  7 16:02:51 2014 mmbackup:Sending files to the TSM server [364 changed, 0 expired].
Mon Apr  7 16:02:51 2014 mmbackup:Performing backup operations
Mon Apr  7 16:05:40 2014 mmbackup:Completed policy backup run with 0 policy errors, 0 files failed, 0 severe
errors, returning rc=0.
Mon Apr  7 16:05:40 2014 mmbackup:Policy for backup returned 0 Highest TSM error 0
```

```
mmbackup: TSM Summary Information:
```

```
Total number of objects inspected:    364
Total number of objects backed up:    364
Total number of objects updated:      0
Total number of objects rebound:     0
Total number of objects deleted:      0
Total number of objects expired:      0
Total number of objects failed:       0
Total number of bytes transferred:    2179695902
```

```
Mon Apr  7 16:05:40 2014 mmbackup:analyzing: results from balok1.
Mon Apr  7 16:05:40 2014 mmbackup:Copying updated shadow file to the TSM server
Mon Apr  7 16:05:44 2014 mmbackup:Done working with files for TSM Server: balok1.
Mon Apr  7 16:05:44 2014 mmbackup:Completed backup and expire jobs.
Mon Apr  7 16:05:44 2014 mmbackup:TSM server balok1
had 0 failures or excluded paths and returned 0.
Its shadow database has been updated. Shadow DB state:updated
Mon Apr  7 16:05:44 2014 mmbackup:Completed successfully.  exit 0
```

mmbackup

```
-----  
mmbackup: Backup of /gpfs/gpfs0 completed successfully at Mon Apr 7 16:05:44 EDT 2014.  
-----
```

4. To perform an incremental backup of the objects in the inode space of the **gpfs/testfs/infs2** directory to the **balok1** server, issue this command:

```
mmbackup /gpfs/testfs/infs2 -t incremental --scope inodespace --tsm-servers balok1
```

The system displays information similar to:

```
-----  
mmbackup: Backup of /gpfs/testfs/infs2 begins at Wed May 27 12:58:39 EDT 2015.  
-----
```

```
Wed May 27 12:58:48 2015 mmbackup:Scanning fileset testfs.indfs2  
Wed May 27 12:58:53 2015 mmbackup:Determining fileset changes for testfs.indfs2 [balok1].  
Wed May 27 12:58:53 2015 mmbackup:changed=2, expired=2, unsupported=0 for server [balok1]  
Wed May 27 12:58:53 2015 mmbackup:Sending files to the TSM server [2 changed, 2 expired].  
mmbackup: TSM Summary Information:  
Total number of objects inspected:      4  
Total number of objects backed up:      2  
Total number of objects updated:        0  
Total number of objects rebound:        0  
Total number of objects deleted:        0  
Total number of objects expired:        2  
Total number of objects failed:         0  
Total number of objects encrypted:      0  
Total number of bytes inspected:        53934  
Total number of bytes transferred:      53995
```

```
-----  
mmbackup: Backup of /gpfs/testfs/infs2 completed successfully at Wed May 27 12:59:31 EDT 2015.  
-----
```

5. To perform an incremental backup of a global snapshot called **backupsnap6** to the **balok1** server, issue this command:

```
mmbackup testfs -t incremental -S backupsnap6 --scope filesystem --tsm-servers balok1
```

The system displays information similar to:

```
-----  
mmbackup: Backup of /gpfs/testfs begins at Wed May 27 13:08:45 EDT 2015.  
-----
```

```
Wed May 27 13:08:50 2015 mmbackup:Scanning file system testfs  
Wed May 27 13:08:53 2015 mmbackup:Determining file system changes for testfs [balok1].  
Wed May 27 13:08:53 2015 mmbackup:changed=130, expired=100, unsupported=0 for server [balok1]  
Wed May 27 13:08:53 2015 mmbackup:Sending files to the TSM server [130 changed, 100 expired].  
  
Wed May 27 13:08:59 2015 mmbackup:Policy for expiry returned 9 Highest TSM error 0  
mmbackup: TSM Summary Information:  
Total number of objects inspected:      230  
Total number of objects backed up:      130  
Total number of objects updated:        0  
Total number of objects rebound:        0  
Total number of objects deleted:        0  
Total number of objects expired:        100  
Total number of objects failed:         0  
Total number of objects encrypted:      0  
Total number of bytes inspected:        151552  
Total number of bytes transferred:      135290
```

```
-----  
mmbackup: Backup of /gpfs/testfs completed successfully at Wed May 27 13:09:05 EDT 2015.  
-----
```

See also

- “mmapplypolicy command” on page 266

Location

/usr/lpp/mmfs/bin

mmbackupconfig command

Collects GPFS file system configuration information.

Synopsis

```
mmbackupconfig Device -o OutputFile
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmbackupconfig** command, in conjunction with the **mmrestoreconfig** command, can be used to collect basic file system configuration information that can later be used to restore the file system. The configuration information backed up by this command includes block size, replication factors, number and size of disks, storage pool layout, filesets and junction points, policy rules, quota information, and a number of other file system attributes.

This command does not back up user data or individual file attributes.

For more information about the **mmimgbackup** and **mmimgrestore** commands, see the topic *Scale out Backup and Restore (SOBAR)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system to be backed up. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-o *OutputFile*

The path name of a file to which the file system information is to be written. This file must be provided as input to the subsequent **mmrestoreconfig** command.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmbackupconfig** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To backup file system **fsiam2** to output file **backup.config.fsiam2** issue:

```
mmbackupconfig fsiam2 -o backup.config.fsiam2
```

The system displays information similar to:


```
mmbackupconfig: Processing file system fsiam2 ...  
mmbackupconfig: Command successfully completed
```

See also

- “mmimgbackup command” on page 508
- “mmimgrestore command” on page 511
- “mmrestoreconfig command” on page 631

Location

/usr/lpp/mmfs/bin

mmbuildgpl

mmbuildgpl command

Manages prerequisite packages for Linux and builds the GPFS portability layer.

Synopsis

```
mmbuildgpl [--quiet] [--build-package] [-v]
```

Availability

Available on all IBM Spectrum Scale editions. Available and needed only on Linux.

Description

Use the **mmbuildgpl** command to manage and verify prerequisite packages for Linux and build the GPFS portability layer. If all packages are installed correctly, **mmbuildgpl** builds the GPFS portability layer. If any packages are missing, the package names are displayed. The missing packages can be installed manually.

Parameters

--quiet

Specifies that when there are any missing packages, the **mmbuildgpl** command installs the prerequisite packages automatically by using the default package manager.

--build-package

Builds an installable package (**gpfs.gplbin**) for the portability layer binaries after compilation is successful. This option builds an RPM package on SLES and RHEL Linux and a Debian package on Debian and Ubuntu Linux.

When the command finishes, it displays the location of the generated package:

```
Wrote: /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-2.6.32-279.e16.x86_64-4.2.0-0.x86_64.rpm
```

or

```
Wrote: /tmp/deb/gpfs.gplbin_4.2.0.0_amd64.deb
```

You can then copy the generated package to other machines for deployment. The generated package can *only* be deployed to machines with identical architecture, distribution level, Linux kernel, and IBM Spectrum Scale maintenance level.

Note: During the package generation, temporary files are written to the **/tmp/rpm** or **/tmp/deb** directory, so be sure there is sufficient space available. By default, the generated package goes to **/usr/src/packages/RPMS/<arch>** for SUSE Linux Enterprise Server, **/usr/src/redhat/RPMS/<arch>** for Red Hat Enterprise Linux, and **/tmp/deb** for Debian and Ubuntu Linux.

-v Specifies that the output is verbose and contains information for debugging purposes.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmbuildgpl** command.

Examples

To build the GPFS portability layer, issue this command:

```
mmbuildgpl
```

The system displays information similar to:

```
-----
mmbuildgpl: Building GPL module begins at Fri Jun 13 16:37:50 EDT 2014.
-----

Verifying Kernel Header...
kernel version = 3001300 (3.0.13-0.27)
kernel module dir = /lib/modules/3.0.13-0.27-default/build/include
kernel source dir = /usr/src/linux-3.0.13-0.27/include
Found a valid kernel include directory: /lib/modules/3.0.13-0.27-default/build/include
Verifying Compiler...
make is present at /usr/bin/make
cpp is present at /usr/bin/cpp
gcc is present at /usr/bin/gcc
g++ is present at /usr/bin/g++
ld is present at /usr/bin/ld
make World ...
make Install ...
-----
mmbuildgpl: Building GPL module completed successfully at Fri Jun 13 16:39:08 EDT 2014.
-----
```

See also

- “Building the GPFS portability layer on Linux nodes”

Location

```
/usr/lpp/mmfs/bin
```

mmcallhome command

Manages the call home operations.

Synopsis

```
mmcallhome group add group server [--node {all | childNode [,childNode...]}]
```

or

```
mmcallhome group list
```

or

```
mmcallhome group delete GroupName
```

or

```
mmcallhome group auto [--server ServerName...]
```

or

```
mmcallhome capability list
```

or

```
mmcallhome capability enable
```

or

mmcallhome

mmcallhome capability disable

or

mmcallhome info list

or

mmcallhome info change --key value [--key value]...

or

mmcallhome proxy enable [--with-proxy-auth]

or

mmcallhome proxy disable

or

mmcallhome proxy list

or

mmcallhome proxy change --key value [--key value]...

or

mmcallhome schedule list

or

mmcallhome schedule add --task { *daily* | *weekly* }

or

mmcallhome schedule delete --task { *daily* | *weekly* }

or

mmcallhome run GatherSend --task { *daily* | *weekly* }

or

mmcallhome run SendFile --file *file* [--desc *text*]

or

mmcallhome status list [--task{ *daily* | *weekly* | **sendfile}] [-n *num*][--verbose]**

or

mmcallhome status delete {--task{ *daily* | *weekly* | **sendfile} | --startTime *time* | --startTimeBefore *time* | --all }**

or

mmcallhome test connection

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmcallhome** command to configure, enable, run, schedule, and monitor call home related tasks in the GPFS cluster. The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

By using this command, predefined data from each node can be collected on a regular basis and uploaded to IBM. IBM support and development teams can use this data to understand how the customers are using IBM Spectrum Scale. In case of issues, the data can be referenced for problem analysis. The data can also possibly be used to provide advice to customers regarding failure prevention.

Since IBM Spectrum Scale consists of multiple nodes, the call home feature introduces call home group to manage them. A call home group consists of one gateway node (which is defined as a call home node) and one or more client nodes (which are defined as call home child nodes). The call home node will initiate the data collection from the call home child nodes and will upload data to IBM using HTTPS protocol. At least one call home group needs to be defined within IBM Spectrum Scale. The call home node needs to have access to the external network via port 443. The maximum number of nodes per group should not exceed 32. Multiple call home groups can be defined within a IBM Spectrum Scale cluster.

For more information about the callhome feature, see "Understanding the call home functionality" in *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

group

Manages topology with one of the following actions:

add

Creates a call home group, which is a group of nodes consisting of one call home node and multiple call home child nodes. Multiple call home groups can be configured within a GPFS cluster.

The call home node initiates data collection within the call home group and uploads the data package to the IBM server.

group

Specifies the name of the call home group.

Note: The group name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', and '.'

server

Specifies the name of the call home server belonging to the call home group.

Note: The server name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', and '.'

--node childNode

Specifies the call home child nodes.

Note: The child node name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', and '.'

--node all

Selects all linux nodes in the GPFS cluster. If the number of nodes exceeds 32, the command will fail. When this parameter is omitted, only the call home node will be added to the child node. Additionally, call home node will be always added to the child node group.

mmcallhome

list

Displays the configured call home groups.

delete

Deletes the specified call home group.

GroupName

Specifies the name of the call home group that needs to be deleted.

auto

Enables automatic creation of a call home group.

--server ServerName

Specifies one or more call home servers. The server must be able to access the call home functionality through internet. If no server is specified, the program detects a server automatically.

If a proxy is needed, specify the proxy by using the **mmcallhome proxy** command.

all

Specifies that each node is a call home server. This configuration avoids heavy work load for scheduled call home tasks on call home servers serving large groups. While using **all** option, it is recommended to use a proxy to access the internet.

Note: Multiple servers can be specified by repeating this option or by specifying a string, containing either a comma or a blank to separate the list of servers. If a group exists then the specified server must not be a member of that group. Each call home server must be able to access the internet either directly or via a proxy.

capability

Manages the overall call home activities with one of the following actions:

list

Displays the configured customer information such as the current enable or disable status, call home node, and call home child nodes.

enable

Enables the call home service.

disable

Disables the currently running call home service.

info

Manages customer data with one of the following actions:

list

Displays the configured parameter values.

change

Sets parameter values.

[--key value]

Indicates a placeholder pointing to the table below.

Table 25. key-value

Key	Value
customer-name	Business/company name This name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', ':', '!', ''

Table 25. key-value (continued)

Key	Value
customer-id	Customer ID This can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', '!'.
email Customer	E-mail ID of the customer. All alphanumeric and non-alphanumeric characters are supported.
country-code 2 alphabet ISO	Country code (Example, US)

proxy

Configures proxy-related parameters with one of the following actions:

enable

Enables proxy access.

[--with-proxy-auth]

Enables user ID and password authentication to the proxy server.

disable

Disables proxy access.

list

Displays the currently configured proxy-related parameter values.

change

Modifies the proxy configuration.

[--key value]

Indicates a placeholder pointing to the table below.

Table 26. key-value

Key	Value
proxy-location	Proxy server address (IP address/fully qualified domain name)
proxy-port	Proxy server port number
proxy-username	Proxy server user name This name can consist of any alphanumeric and non-alphanumeric characters.
proxy-password	Proxy server password This can consist of any alphanumeric and non-alphanumeric characters.

schedule

Configures scheduling of call home tasks with one of the following actions:

list

Displays the registered gather-send tasks. A gather-send task is a process that runs on the call home node to collect data from the child nodes and upload the data to the configured server. The gather-end configuration file will include information about what needs to be collected from the child nodes.

add

Registers the specified task to cron.

--task daily

Specifies the configuration file for the daily task.

mmcallhome

--task weekly
Specifies the configuration file for the weekly task.

delete

Removes a daily or weekly task from cron with one of the following options:

--task daily
Specifies the daily task that needs to be removed from cron.

--task weekly
Specifies the weekly task that needs to be removed from cron.

run

Executes one-time gather or send tasks with one of the following options:

GatherSend

Executes one-time gather-send task to collect data and upload.

--task daily
Specifies the daily task that needs to be executed.

--task weekly
Specifies the weekly task that needs to be executed.

SendFile

Uploads a specified file to IBM, with the following options:

--file file
Specifies the name of the file that needs to be uploaded.

Note: The name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', '.'

[--desc text]
Specifies the description of the file that needs to be uploaded. This will be added to the data package file name.

Note: This text can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', '.', ':', ' ', ''

status

Displays status of the call home tasks with one of the following options:

list

Displays the status of the currently running and the already completed call home tasks.

--task [daily | weekly]
Specifies the log of the daily or weekly task.

--task sendfile
Specifies the status of the tasks initiated by the "run sendfile" command.

[-n num]
Specifies the number of entry per gather-send task to show.

--verbose
Specifies additional information.

When the **mmcallhome list --verbose** command is executed, the following information will be shown in the output:

- **Task:** Name of the gather-send configuration file.
- **Started time:** Timestamp when the gather-send task is invoked.
- **Updated time:** Timestamp when the status is updated.

- **Status:** Success/minor error/failed/running.
- **RC or Step:** When the status is failed/success, the return code of the task is shown. See below for the return code description of the gather-send task. When the status is running, the step is shown. See below for the step description.
 - **Package file name:** Name of the created data package to upload.
 - **additional info:** Any additional info for the task.

Gather-send task return codes:

- 0 - Success
- 1 - Successfully uploaded after a few send retries
- 2 - Some gather commands failed but logs collected from all child nodes and successfully uploaded
- 3 - Could not collect logs from some nodes but the call home node created the data package and successfully uploaded
- 4 - Error in command parameter
- 5 - Call home is disabled
- 6 - Another gather-send task for the same configuration file is already running
- 7 - Error in gather-send configuration file
- 8 - Data package created but sender failed
- 9 - Internal error
- 10 - Critical error
- 99 - Unknown

Gather-send task steps:

- step 1 - Initializing
- step 2 - Each call home child nodes gathering logs
- step 3 - Pulling log collection from child nodes
- step 4 - Creating data package
- step 5 - Calling send task
- step 6 - Final status

delete

Deletes the status log of the specified configuration file with the following options:

--task [daily | weekly]

Specifies the log of the daily or weekly task.

--task sendfile

Specifies the log of the tasks initiated by the "run sendfile" command.

[-n num]

Specifies the number of entry per gather-send task to show.

--startTime starttime

Specifies the start time of the log to delete.

--startTimeBefore starttime

All logs older than the time specified by this option will be deleted..

--all

All logs will be deleted.

test

Executes a system check to ensure that a connection is established:

connection

Specifies the connection to check.

mmcallhome

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcallhome** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To configure a call home group, issue this command:

```
mmcallhome group add group1 themisto0 -N themisto0,themisto1,themisto2
```

The system displays output similar to this:

```
Call home group group1 has been created
```

2. To view the configured call home groups, issue this command:

```
mmcallhome group list
```

The system displays output similar to this:

Call Home Group	Call Home Node	Call Home Child Nodes
group1	themisto0	themisto0,themisto1,themisto2

3. To change customer information such as customer name, customer ID, and the country code, issue this command:

```
mmcallhome info change --customer-name "SpectrumScaleTest" --customerid "1234" --country-code "JP"
```

The system displays output similar to this:

```
Success
```

4. To view the customer information, issue this command:

```
mmcallhome info list
```

The system displays output similar to this:

Parameter	Value
customer-name	SpectrumScaleTest
customer-id	1234
voice-phone	
offshift-phone	
modem-phone	
modem-prefix	
ign1phone	
ign2phone	
customer-location	
email	
special-instruction	
callhome-method	ethernet
remote-service-callback-number	
country-code	JP

5. To enable the call home service, issue this command:

```
mmcallhome capability enable
```

The system displays output similar to this:

```
Call home node: themisto0
Call home child nodes to collect data: themisto0 themisto1 themisto2 (total 3 nodes)
Excluded nodes:
SSH Access Check: OK
Data package directory: /tmp/mmfs/callhome
Success
```

- To register a daily task with cron, issue this command:

```
mmcallhome schedule add --task daily
```

The system displays output similar to this:

```
/etc/cron.d/gpfs_callhome_gatherSend_daily.conf registered
41 command entries are defined for this task
```

- To register a weekly task with cron, issue this command:

```
mmcallhome schedule add --task weekly
```

The system displays output similar to this:

```
/etc/cron.d/gpfs_callhome_gatherSend_weekly.conf registered
14 command entries are defined for this task
```

- To list the registered tasks for gather-send, issue this command:

```
mmcallhome schedule list
```

The system displays output similar to this:

```
Registered Tasks for gatherSend:
ConfFile      CronParameters
daily.conf    3 2 * * *
weekly.conf   54 3 * * sun
```

- To monitor the call home tasks, issue this command:

```
mmcallhome status list
```

The system displays output similar to this:

```
Task   Start time      Status      Package file name
daily  20150930132656.582  success    ...aultDaily.g_daily.20150930132656582.c10.DC
daily  20150930133134.802  success    ...aultDaily.g_daily.20150930133134802.c10.DC
daily  20150930133537.509  success    ...aultDaily.g_daily.20150930133537509.c10.DC
daily  20150930133923.063  success    ...aultDaily.g_daily.20150930133923063.c10.DC
RunSendFile 20150930133422.843 success
...group2.MyTestData.s_file.20150930133422843.c10.DC
```

- To set the parameters for the proxy server, issue this command:

```
mmcallhome proxy change --proxy-location okapi --proxy-port 80 --proxyusername
root --proxy-password <password>
```

The system displays output similar to this:

```
Success
```

- To view the proxy server parameters, issue this command:

```
mmcallhome proxy list
```

The system displays output similar to this:

```
Status
proxy-enabled      NO
proxy-auth-enabled false
Parameter
proxy-location     okapi
proxy-port 80
proxy-username     root
proxy-password     xxxxx
```

mmcallhome

12. To invoke a one-time gather-send task, issue this command:

```
mmcallhome run GatherSend --task daily
```

The system displays output similar to this:

```
Starting one time run using daily.conf
```

13. To run one-time send command to upload a file, issue this command:

```
mmcallhome run SendFile --file /ibm/gpfs0/testDir/testFile --desc MyTestData
```

The system displays output similar to this:

```
Running sendFile... (In case of network errors, it may take over 20 minutes for  
retries.)
```

```
StartTime=20150930193046.693
```

```
Successfully uploaded the given file
```

```
Run mmcallhome status ls -v to see the package name
```

14. To view the status of the currently running and the already completed call home tasks, issue this command:

```
mmcallhome status list --verbose
```

The system displays output similar to this:

```
Task Start time Updated time Status RC or Step
```

```
Package file name
```

```
[ additional info: value ]
```

```
-----  
RunSendFile 20150930193046.693 20150930193059 success RC=0
```

```
11786648094375.4_2_0_0.1234.SpectrumScaleTest.group1.MyTestData.s_fil
```

```
e.20150930193046693.c10.DC
```

```
Original file name: testFile  
-----
```

15. To test the connection, issue this command:

```
mmcallhome test connection
```

The system displays output similar to this:

```
Starting connectivity test between the call home node and IBM
```

```
Call home node: themisto0
```

```
Starting time: Wed Sep 30 14:37:51 JST 2015
```

```
Testing connection via proxy server (no authentication required)
```

```
User: NA Pass: NA Host: okapi Port: 80
```

```
Testing prefix Edge_SP_Config:
```

```
Edge_SP_Config_1: 129.42.56.189 OK
```

```
Testing prefix Edge_Profile:
```

```
Edge_Profile_1: 129.42.56.189 OK
```

```
Testing prefix Edge_Status_Report:
```

```
Edge_Status_Report_1: 129.42.56.189 OK
```

16. To check the version and the subversion of the mmcallhome program, issue this command:

```
mmcallhome version
```

The system displays output similar to this:

```
Version: 4.2.0-005
```

See also

- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmobj command” on page 581

- “mmsmb command” on page 663
- “mmuserauth command” on page 690

Location

/usr/lpp/mmfs/bin

mmces command

Manages CES configuration.

Synopsis

```
mmces address add [--ces-node Node] [--attribute Attribute] [--ces-group Group] [--ces-ip {IP[,IP...]}
```

or

```
mmces address remove --ces-ip {IP[,IP...]}
```

or

```
mmces address move --ces-ip {IP[,IP...]} --ces-node Node
```

or

```
mmces address move --rebalance
```

or

```
mmces address change
    [--ces-ip IP | --remove-attribute]
    --attribute Attribute[,Attribute...]
```

```
mmces address change
    [--ces-ip IP[,IP...]]
    [--attribute Attribute[,Attribute...]]
    [--ces-group Group]
    [--remove-attribute]
    [--remove-group]
```

or

```
mmces address list [-N {Node[,Node...]} | NodeFile | NodeClass]
mmces address list [--ces-ip IP[,IP...]] [--ces-group Group[,Group...]]
    [--attribute Attribute[,Attribute...]] [--by-node]
```

or

```
mmces address policy [none | balanced-load | node-affinity | even-coverage]
```

or

```
mmces node {suspend | resume} [-N {Node[,Node...]} | NodeFile | NodeClass} | -a]
```

or

```
mmces node list [--ces-group Group[,Group...]] [--verbose]
```

or

```
mmces service {enable | disable} {NFS | SMB | OBJ | BLOCK}
```

or

```
mmces service {start | stop} {NFS | SMB | OBJ | BLOCK}
    [-N {Node[,Node...]} | NodeFile | NodeClass} | -a]
```

or

```
mmces service list [-N {Node[,Node...]} | NodeFile | NodeClass} | -a] [--verbose]
```

or

```
mmces log level [new-level]
```

or

```
mmces events active [NFS | OBJ | SMB | AUTH | NETWORK]
                   [-N {Node[,Node...]} | NodeFile | NodeClass} | -a]
```

or

```
mmces events list [NFS | OBJ | SMB | AUTH | NETWORK]
                  [--time {hour | day | week | month}]
                  [--severity {INFO | WARNING | ERROR | SEVERE}]
                  [-N {Node[,Node...]} | NodeFile | NodeClass} | -a]
```

or

```
mmces state show [NFS | OBJ | SMB | AUTH | NETWORK | CES]
                 [-N {Node[,Node...]} | NodeFile | NodeClass} | -a]
```

or

```
mmces state cluster [NFS | OBJ | SMB | AUTH | AUTH_OBJ | NETWORK | CES]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmces** command to manage protocol addresses, services, node state, logging level and load balancing. The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

CES currently supports the NFS, SMB, BLOCK, and Object services. Each service can be enabled or disabled with the **mmces service** command. Enabling a service is a CES cluster-wide operation. In addition, enabled services can be started and stopped on individual nodes.

Clients access the CES services using one or more IP addresses in the CES address pool. Addresses can be added to and removed from the pool using the **mmces address add** and **mmces address remove** commands. Existing addresses can be reassigned to another node with the **mmces address move** command.

Addresses can have one or more attributes associated with them. An address attribute is a tag that the services can identify a specific address as having a special meaning, which is defined by the service protocol. Addresses can have multiple attributes, but an attribute can only be associated with a single address.

Addresses can have a policy associated with them, and that policy determines how addresses are automatically distributed. The allowed policies are **none**, **balanced-load**, **node-affinity**, and **even-coverage**. A policy of **none** means addresses are not distributed automatically.

A CES node can be placed in a suspended state. When a node is in suspended state, all of the CES addresses for that node are reassigned to other nodes, and the node will not accept new address assignments. Any services that are started when the node is suspended continue to run. The suspended state is persistent, which means nodes remain suspended following a reboot. Use the **mmces node** command to suspend and resume a node.

Parameters

address

Manages CES addresses with one of the following actions:

mmces

add

Adds the addresses specified by the **--ces-ip** parameter to the CES address pool and assigns them to a node. The node to which an address is assigned will configure its network interfaces to accept network communication destined for the address. CES addresses must be different from IP addresses used for GPFS or CNFS communication.

If **--ces-node** is specified with **add**, all addresses specified with the **--ces-ip** parameter will be assigned to this node. If **--ces-node** is not specified, the addresses will be distributed among the CES nodes.

If an attribute is specified with **--attribute**, there can only be one address specified with the **--ces-ip** parameter.

If **--ces-group** is specified with **add**, all new addresses will be associated with the specified group. The result can be viewed with the "mmces address list" command.

remove

Removes the addresses specified by the **--ces-ip** parameter from the CES address pool. The node to which the address is assigned will reconfigure its network interfaces to no longer accept communication destined for the address. The node specified in the command will configure its network interfaces to accept communication for this address.

move

Moves addresses.

If the **--ces-ip** parameter is specified, the addresses specified by *IP* are moved from one CES node to another. The addresses are reassigned to the node specified by the **--ces-node** parameter.

If the **--rebalance** parameter is specified, the addresses are distributed immediately based on the currently configured distribution policy. If the policy is currently undefined or **none**, the **even-coverage** policy is applied.

Use this command with caution because IP movement will trigger CES protocol recovery.

change

Changes or removes address attributes.

If the **--ces-ip** parameter is specified:

- the attributes specified by the **--attribute** parameter are associated with the address specified by the **--ces-ip** parameter. If an attribute was previously associated with another address, that association is ended.
- the attributes specified by the **--attribute** parameter, with the **--remove-attribute** parameter, have the specified attributes removed from the addresses specified by the **--ces-ip** parameter.
- groups specified by the **--ces-group** parameter are associated with the address specified by the **--ces-ip** parameter.
- groups specified by the **--ces-group** parameter, with the **--remove-group** parameter, have the specified groups removed from the addresses specified by the **--ces-ip** parameter and is no longer used.

If the **--ces-ip** parameter is not specified:

- specifying **--remove-attribute**, with the attributes specified by the attributes specified by the **--attribute** parameter, removes the attributes from their current associations and they are no longer used.
- specifying **--remove-group**, with the groups *s* specified by the **--group** parameter, removes the groups from their current associations and they are no longer used.

list

Lists the CES addresses along with group, attribute and node assignments.

Options:

- **--ces-ip** Only list the addresses provided.

- ces-group** Only list addresses whose group assignment matches one of the groups provided.
- attribute** Only list addresses whose attributes match one of the attributes provided.
- ces-node** Only list addresses assigned to one of the nodes provided.
- by-node** List addresses by node, using the output format from IBM Spectrum Scale V4.1.1 and later.

policy

Sets the CES address distribution policy.

node

Manages CES node state with one of the following actions:

suspend

Suspends the specified nodes. If neither the **-N** or **-a** parameters are specified, only the local node is suspended.

When a node is suspended, all addresses assigned to the node are reassigned to other nodes and the node will not accept any subsequent address assignments. Suspending a node will trigger CES protocol recovery if the node has CES addresses assigned.

resume

Resumes the specified suspended nodes. If neither the **-N** or **-a** parameters are specified, only the local node is resumed.

When a suspended node is resumed (no longer suspended), the node will accept subsequent address assignments.

list

Lists the specified nodes along with their current node state. If the **-N** parameter is not specified, all nodes are listed.

verbose

Lists the addresses assigned to the nodes.

--ces-group

Lists the nodes belonging to the specified groups.

service

Manages protocol services with one of the following actions:

enable

Enables and starts the specified service on all CES nodes.

disable

Disables and stops the specified service on all CES nodes.

Note: Disabling a service will discard any configuration data from the CES cluster and needs to be used with caution. If applicable, backup any relevant configuration data. Subsequent service enablement will start with a clean configuration.

start

Starts the specified service on the nodes specified. If neither the **-N** or **-a** parameters are specified, the service is started on the local node.

stop

Stops the specified service on the nodes specified. If neither the **-N** or **-a** parameters are specified, the service is stopped on the local node.

Note: If a service is stopped on a node that has CES addresses assigned, clients will not be able to access the service using any of the addresses assigned to that node. Access to the data from clients is not possible any more for services that are stopped.

mmces

list

Lists the state of the enabled services.

log level

Sets and checks the CES log level. The CES log level determines how much information related to the CES nodes is entered into the GPFS log file. Values can be from 0 (less logging) to 3 (more logging).

events

Shows one of the following CES events that occurred on a node or nodes:

active

Lists all events that are currently contributing to making the state of a component unhealthy. If no component is specified, active events for all components are listed. If neither the **-N** or **-a** parameters are specified, the active events for the local node are listed. If there are multiple events shown by the command they will be listed in the order we recommend they be fixed, with the most important event to fix at the top.

list

Lists the events that occurred on a node or nodes, whether or not they are currently contributing to the state of a component. If no component is specified, events for all components are listed. If **--time** is specified, only events from the previous chosen interval are listed, otherwise all events are listed. If **--severity** is specified, only events of the chosen severity are listed, otherwise all events are listed. If neither the **-N** or **-a** parameters are specified, the events for the local node are listed.

Events older than 180 days are removed from the list. A maximum of 10,000 events are saved in the list.

state

Shows the state of one or more nodes in the cluster.

show

Shows the state of the specified service on the nodes specified. If no service is specified, all services will be displayed. If neither the **-N** or **-a** parameters are specified, the state of the local node is shown.

cluster

Shows the combined state for the services across the whole CES cluster. If no service is specified an aggregated state will be displayed for each service, where healthy means the service is healthy on all nodes, degraded means the service is not healthy on one or all nodes, and failed means that the service is not available on any node. If a service is specified the state of that service will be listed for each node, along with the name of any event that is contributing to an unhealthy state.

--ces-node

Indicates that the command applies only to the specified CES node name.

--attribute

Specifies either a single attribute or a comma-separated list of attributes as indicated in the command syntax.

--ces-ip

Specifies either a single or comma-separated list of DNS qualified host names or IP addresses as indicated in the command syntax.

--rebalance

Distributes addresses immediately based on the currently configured distribution policy. If the policy is currently undefined or **none**, the **even-coverage** policy is applied.

none

Specifies that addresses are not distributed automatically.

balanced-load

Distributes addresses dynamically in order to approach an optimized load distribution throughout the cluster. The network and CPU load on all the nodes is monitored and addresses are moved based on given policies.

Addresses that were recently moved or addresses with attributes are not moved.

node-affinity

Attempts to keep addresses associated with the node to which they were assigned. Address node associations are created with the `--ces-node` parameter of the **mmces address add** command or the **mmces address move** command. Automatic movements of addresses do not change the association. Addresses that were enabled without a node specification do not have a node association. Addresses that are associated with a node but assigned to a different node are moved back to the associated node.

Addresses that were recently moved or addresses with attributes are not moved.

even-coverage

Attempts to evenly distribute all of the addresses among the available nodes.

Addresses that were recently moved or addresses with attributes are not moved.

--remove-attribute

Indicates that the specified attributes should be removed.

-N {Node[,Node...] | NodeFile | NodeClass}

Indicates that the command applies only to the specified node names.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-a Specifies that the command applies to all CES nodes.

NFS

Specifies that the command applies to the NFS service.

OBJ

Specifies that the command applies to the Object service.

SMB

Specifies that the command applies to the SMB service.

AUTH

Specifies that the command applies to the AUTH service.

NETWORK

Specifies that the command applies to the NETWORK service.

CES

Specifies that the command applies to the CES service.

--verbose

Specifies that the output is verbose.

new-level

Sets the log level to a new value. If the *new-level* parameter is not specified, the current log level is displayed.

--time

Lists the previous events from one of the following intervals:

hour

Lists the events from the past hour.

day

Lists the events from the past day.

mmces

week

Lists the events from the past week.

month

Lists the events from the past month.

The events are listed whether or not they are currently contributing to the state of a component.

--severity

Specifies that only events for one of the following severities are listed:

INFO

Lists only informational events.

WARNING

Lists only warning events.

ERROR

Lists only error events.

SEVERE

Lists only severe events.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmces** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To add an address to a specified node, issue this command:

```
mmces address add --ces-node node1 --ces-ip 10.1.2.3
```

When this command is successful, the system does not display output.

2. To add several addresses to a specified node, issue this command:

```
mmces address add --ces-node node1 --ces-ip 10.1.2.3,10.1.2.4
```

When this command is successful, the system does not display output.

3. To add an address with the attribute **xyz_server** to a specified node, issue this command:

```
mmces address add --ces-node node1 --ces-ip 10.1.2.3 --attribute xyz_server
```

When this command is successful, the system does not display output.

4. To add addresses which are distributed among the CES nodes, issue this command:

```
mmces address add --ces-ip 10.1.2.3,10.1.2.4,10.1.2.5,10.1.2.6
```

When this command is successful, the system does not display output.

5. To remove several addresses, issue this command:

```
mmces address remove --ces-ip 10.1.2.3,10.1.2.4
```

When this command is successful, the system does not display output.

- To associate the attribute **xyz_server** to the address **10.1.2.3**, issue this command:

```
mmces address change --ces-ip 10.1.2.3 --attribute xyz_server
```

When this command is successful, the system does not display output.

- To remove the attribute **xyz_server**, issue this command:

```
mmces address change --remove-attribute --attribute xyz_server
```

When this command is successful, the system does not display output.

- To move an address to another node, issue this command:

```
mmces address move --ces-ip 10.0.100.231 --ces-node node2
```

When this command is successful, the system does not display output.

- To suspend a group of nodes, issue this command:

```
mmces node suspend -N node1,node2,node3
```

The system displays output similar to this:

```
Node now in suspended state.
```

- To resume the current node, issue this command:

```
mmces node resume
```

The system displays output similar to this:

```
Node no longer in suspended state.
```

- To enable the Object service in the CES cluster, issue this command:

```
mmces service enable obj
```

When this command is successful, the system does not display output.

- To disable the NFS service in the CES cluster, issue this command:

```
mmces service disable nfs
```

When this command is successful, the system does not display output.

- To stop the SMB service on a few nodes, issue this command:

```
mmces service stop smb -N node1,node2,node3
```

When this command is successful, the system does not display output.

- To start the SMB service on all CES nodes, issue this command:

```
mmces service start smb -a
```

When this command is successful, the system does not display output.

- To show which services are enabled and which are running all CES nodes, issue this command:

```
mmces service list -a
```

The system displays output similar to this:

```
Enabled services: NFS OBJ
node1: NFS is running, OBJ is running
node2: NFS is running, OBJ is running
node3: NFS is running, OBJ is running
```

- To display the current CES log level, issue this command:

```
mmces log level
```

The system displays output similar to this:

mmces

CES log level is currently set to 1

17. To set the CES log level to 2, issue this command:

```
mmces log level 2
```

The system displays output similar to this:

```
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

18. To display the state of all CES components on the local node, issue this command:

```
mmces state show
```

The system displays output similar to this:

NODE	AUTH	AUTH_OBJ	NETWORK	NFS	OBJ	SMB	CES
node1	HEALTHY	DISABLED	HEALTHY	HEALTHY	HEALTHY	DISABLED	HEALTHY

19. To display the state of the NFS component on all nodes, issue this command:

```
mmces state cluster NFS
```

The system displays output similar to this:

NODE	COMPONENT	STATE	EVENTS
node1	NFS	HEALTHY	
node2	NFS	FAILED	nfsd_down
node3	NFS	HEALTHY	

20. To display a list of active events of all CES components on the local node, issue this command:

```
mmces events active
```

The system displays output similar to this:

Node	Component	Event Name	Severity	Details
node1	NFS	nfsd_down	ERROR	NFSD process not running

21. To display a list of all NFS events from the last hour on the local node, issue this command:

```
mmces events list
```

The system displays output similar to this:

Node	Timestamp	Event Name	Severity	Details
node1	2015-05-13 10:57:52.124369+00:00UTC	nfsd_down	ERROR	NFSD process not running
node1	2015-05-13 10:58:06.809071+00:00UTC	cesnodestatechange_info	INFO	A CES node state changed
node1	2015-05-13 10:58:07.137343+00:00U	ganeshagrace_info	INFO	Ganesha NFS is set to grace

See also

- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmobj command” on page 581
- “mmsmb command” on page 663
- “mmuserauth command” on page 690

Location

```
/usr/lpp/mmfs/bin
```

mmcesdr command

Manages protocol cluster disaster recovery.

Synopsis

```
mmcesdr primary config --output-file-path FilePath --ip-list IPAddress [,IPAddress,...]
                        [--allowed-nfs-clients {--all | --gateway-nodes | IPAddress [,IPAddress,...]}]
                        [--rpo RPOValue] [--inband] [-v]
```

or

```
mmcesdr primary backup [-v]
```

or

```
mmcesdr primary restore [--new-primary] [--input-file-path FilePath] [--file-config {--recreate | --restore}] [-v]
```

or

```
mmcesdr primary update {--obj | --nfs | --smb | --ces} [-v]
```

or

```
mmcesdr primary failback --prep-outband-transfer --input-file-path FilePath [-v]
```

or

```
mmcesdr primary failback --convert-new --output-file-path FilePath --input-file-path FilePath [-v]
```

or

```
mmcesdr primary failback {--start | --apply-updates | --stop [--force]} [--input-file-path FilePath] [-v]
```

or

```
mmcesdr secondary config --input-file-path FilePath [--prep-outband-transfer] [--inband] [-v]
```

or

```
mmcesdr secondary failover [--input-file-path FilePath]
                           [--file-config {--recreate | --restore}] [-v]
```

or

```
mmcesdr secondary failback --generate-recovery-snapshots --output-file-path FilePath
                           [--input-file-path FilePath] [-v]
```

or

```
mmcesdr secondary failback --post-failback-complete [--input-file-path FilePath]
                           [--file-config {--recreate | --restore}] [-v]
```

or

```
mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path FilePath
                           [--file-config {--recreate | --restore}] [-v]
```

Availability

Available with IBM Spectrum Scale Advanced Edition.

Description

Use the **mmcesdr** command to manage protocol cluster disaster recovery.

mmcesdr

You can use the **mmcesdr primary config** command to perform initial configuration for protocols disaster recovery on the primary cluster and to generate a configuration file that is used on the secondary cluster. The protocol configuration data can be backed up using the **mmcesdr primary backup** command and the backed up data can be restored using the **mmcesdr primary restore** command. The backed up configuration information for the primary cluster can be updated using the **mmcesdr primary update** command. You can use the **mmcesdr primary failback** command to fail back the client operations to the primary cluster.

You can use the **mmcesdr secondary config** command to perform initial configuration for protocols disaster recovery on the secondary cluster using the configuration file generated from the primary cluster. The secondary read-only filesets can be converted into read-write primary filesets using the **mmcesdr secondary failover** command. You can use the **mmcesdr secondary failback** command to either generate a snapshot for each acting primary fileset or complete the failback process, and convert the acting primary filesets on the secondary cluster back into secondary filesets.

For information on detailed steps for protocols disaster recovery, see *Protocols disaster recovery* in *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

primary

This command is run on the primary cluster.

config

Perform initial configuration of protocol cluster disaster recovery.

--output-file-path *FilePath*

Specifies the path to store output of the generated configuration file, which is always named DR_Config.

--ip-list *IPAddress[,IPAddress,...]*

Comma separated list of public IP addresses on the secondary cluster to be used for active file management (AFM) DR related NFS exports.

--allowed-nfs-clients {*--all* | *--gateway-nodes* | *IPAddress[,IPAddress,...]*}

Optional. Specifies the entities that can connect to the AFM DR related NFS shares, where:

--all

Specifies that all clients must be allowed to connect to the AFM DR related NFS shares. If omitted, the default value of *--all* is used.

--gateway-nodes

Specifies the gateway nodes currently defined on the primary that must be allowed to connect to the AFM DR related NFS shares.

IPAddress[,IPAddress,...]

Specifies the comma separated list of IP addresses that must be allowed to connect to the AFM DR related NFS shares.

--rpo *RPOvalue*

Optional. Specifies the integer value of recovery point objective (RPO) to use for AFM DR filesets. If omitted, the default value of 15 is used. The valid range is: $5 \leq RPO \leq 2147483647$.

--inband

Optional. Specifies to use the inband (across the WAN) method of initial data transfer from primary to secondary cluster. If omitted, the default value of outband is used.

backup

Backs up all protocol configuration and CES configuration into a dedicated, independent fileset with each protocol in its own subdirectory.

restore

Restores object, NFS, and SMB protocol configuration and CES configuration from the configuration data backed up.

--new-primary

Optional. Performs restore operation to a newly, failed back primary cluster.

--input-file-path *FilePath*

Optional. Specifies the original configuration file that was used to set up the secondary cluster. If not specified, the file saved in the configuration independent fileset will be used as default.

--file-config {**--recreate** | **--restore**}

Optional. Specifies whether SMB and NFS exports will be re-created, or if the entire protocol configuration will be restored. If not specified, the SMB and NFS exports will be re-created by default.

update

Updates the backed up copy of the protocol configuration or CES configuration.

--obj

Specifies the backed up copy of the object protocol configuration to be updated with the current object configuration.

--nfs

Specifies the backed up copy of the IBM NFSv4 stack protocol configuration to be updated with the current IBM NFSv4 stack configuration.

--smb

Specifies the backed up copy of the SMB protocol configuration to be updated with the current SMB configuration.

--ces

Specifies the backed up copy of the CES configuration to be updated with the current CES configuration.

failback

Used for several options to failback client operations to a primary cluster.

Failback involves transfer of data from the acting primary (secondary) cluster to the old primary cluster as well as restoring protocol and possibly CES configuration information and transformation of protected filesets to primary filesets.

--prep-outband-transfer

Creates independent filesets that out of band data will be transferred to.

--input-file-path *FilePath*

Specifies the configuration file that is the output from the **mmcesdr secondary failback --generate-recovery-snapshots** command.

failback

Used for several options to failback client operations to a primary cluster.

Failback involves transfer of data from the acting primary (secondary) cluster to the old primary cluster as well as restoring protocol and possibly CES configuration information and transformation of protected filesets to primary filesets.

--convert-new

Specifies that the failback is not going to the old primary but instead a new primary and this step specifically converts the newly created independent filesets to primary AFM DR filesets.

mmcesdr

--output-file-path *FilePath*

Specifies the path to store output of generated configuration file, DR_Config, with the new AFM primary IDs.

--input-file-path *FilePath*

Specifies the configuration file that is the output from the **mmcesdr secondary failback --generate-recovery-snapshots** command.

failback

Used for several options to failback client operations to a primary cluster.

Failback involves transfer of data from the acting primary (secondary) cluster to the old primary cluster as well as restoring protocol and possibly CES configuration information and transformation of protected filesets to primary filesets.

--start

Begins the failback process and restores the data to the last RPO snapshot.

--apply-updates

Transfers data that has been written to the secondary cluster while failover was in-place.

Note: The use of this option might need to be done more than once depending on the system load.

--stop [--force]

Completes the transfer of data process and puts the filesets in the read-write mode. Optionally, if this fails you can use the **--force** option.

Note: In addition to using these options, after stopping the data transfer, you need to use the **mmcesdr primary restore** command to restore the protocol and the CES configuration.

--input-file-path *FilePath*

Optional. Specifies the original configuration file that was used to set up the secondary cluster. If not provided, the default will be to use the one saved in the configuration independent fileset.

secondary

This command is run on the secondary cluster.

config

Perform initial configuration of protocol cluster disaster recovery.

--prep-outband-transfer

Creates independent filesets that out of band data will be transferred to as part of the initial configuration. If out of band data transfer is used for DR configuration, this option must be used before data is transferred from the primary to the secondary using out of band transfer. If out of band transfer is used, this command is run once with this option and then again after the data is transferred without the option.

--input-file-path *FilePath*

Specifies the path of the configuration file generated from the configuration step of the primary cluster.

--inband

Optional. Specifies to use the inband (across the WAN) method of initial data transfer from primary to secondary cluster. If omitted, the default value of outband is used.

Note: If **--inband** is used for the primary configuration, it must also be used for the secondary configuration.

failover

Converts secondary filesets from read-only to read-write primary filesets and converts the secondary protocol configurations to those of the failed primary.

--input-file-path *FilePath*

Optional. Specifies the original configuration file that was used to set up the secondary cluster. If not specified, the file saved in the configuration independent fileset will be used as default.

--file-config {**--recreate** | **--restore**}

Optional. Specifies whether SMB and NFS exports will be re-created, or if the entire protocol configuration will be restored. If not specified, the SMB and NFS exports will just be re-created by default.

failback

Runs one of the two failback options: either generates a snapshot for each acting primary fileset or completes the failback process and convert the acting primary filesets on the secondary cluster back into secondary filesets

--generate-recovery-snapshots

Generates the psnap0 snapshot for each acting primary fileset and stores in the default snapshot location for use in creation of a new primary cluster with new primary filesets to fail back to. The files within the snapshot need to be manually transported to the new primary.

--output-file-path *FilePath*

Specifies the path to store output of generated snapshot recovery configuration file.

--input-file-path *FilePath*

Optional. Specifies the path of the original configuration file that was used to set up the secondary cluster. If not provided, the default is to use the one saved in the configuration independent fileset.

failback

Runs one of the two failback options: either generates a snapshot for each acting primary fileset or completes the failback process and convert the acting primary filesets on the secondary cluster back into secondary filesets

--post-failback-complete

Completes the failback process by converting the acting primary filesets back into secondary, read-only filesets and ensures that the proper NFS exports for AFM DR exist.

--new-primary

Performs the failback operation to a newly, failed back primary cluster.

--input-file-path *FilePath*

Specifies the path of the updated configuration file created from the **mmcesdr primary failback --convert-new** command which includes updated AFM primary IDs.

--file-config {**--recreate** | **--restore**}

Optional. Specifies whether SMB and NFS exports will be re-created, or if the entire protocol configuration will be restored. If not specified, the SMB and NFS exports will just be re-created by default.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmcesdr

Security

You must have root authority to run the **mmcesdr** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See the following *IBM Spectrum Scale: Administration and Programming Reference* topic: "Requirements for administering a GPFS file system" on page 1.

Examples

1. Issue the following command on the primary cluster to configure independent fileset exports as AFM DR filesets and backup configuration information:

```
mmcesdr primary config --output-file-path /root/ --ip-list "9.11.102.211,9.11.102.210" --rpo 10 --inband
```

The system displays output similar to this:

```
Performing step 1/5, configuration fileset creation/verification.
Successfully completed step 1/5, configuration fileset creation/verification.
Performing step 2/5, protocol and export services configuration backup.
Successfully completed step 2/5, protocol and export services configuration backup.
Performing step 3/5, determination of protocol exports to protect with AFM DR.
WARNING: Export /gpfs/fs0/nfs-ganesha-dep of type nfs will NOT be protected through AFM DR because it is a dependent fileset.
Not all exports of type NFS-ganesha will be protected through AFM DR, rc: 2
WARNING: Export /gpfs/fs0/smb-dep of type smb will NOT be protected through AFM DR because it is a dependent fileset.
Not all exports of type SMB will be protected through AFM DR, rc: 2
Completed with errors step 3/5, determination of protocol exports to protect with AFM DR.
Performing step 4/5, conversion of protected filesets into AFM DR primary filesets.
Successfully completed step 4/5, conversion of protected filesets into AFM DR primary filesets.
Performing step 5/5, creation of output DR configuration file.
Successfully completed step 5/5, creation of output DR configuration file.
```

File to be used with secondary cluster in next step of cluster DR setup: /root//DR_Config

2. Issue the following command on the secondary cluster to create the independent filesets that will be a part of the pair of AFM DR filesets associated with those on the primary cluster:

```
mmcesdr secondary config --input-file-path /root/ --inband
```

In addition to fileset creation, this command also creates the necessary NFS exports and converts the independent filesets to AFM DR secondary filesets. The system displays output similar to this:

```
Performing step 1/3, creation of independent filesets to be used for AFM DR.
Successfully completed step 1/3, creation of independent filesets to be used for AFM DR.
Performing step 2/3, creation of NFS exports to be used for AFM DR.
Successfully completed step 2/3, creation of NFS exports to be used for AFM DR.
Performing step 3/3, conversion of independent filesets to AFM DR secondary filesets.
Successfully completed step 3/3, conversion of independent filesets to AFM DR secondary filesets.
```

3. Issue the following command on the primary cluster to configure independent fileset exports as AFM DR filesets, back up configuration information, and facilitate outband data transfer (which is the default):

```
mmcesdr primary config --output-file-path /root/ --ip-list "9.11.102.211,9.11.102.210" --rpo 10
```

The system displays output similar to this:

```
Performing step 1/5, configuration fileset creation/verification.
Successfully completed step 1/5, configuration fileset creation/verification.
Performing step 2/5, protocol and export services configuration backup.
Successfully completed step 2/5, protocol and export services configuration backup.
Performing step 3/5, determination of protocol exports to protect with AFM DR.
Successfully completed step 3/5, determination of protocol exports to protect with AFM DR.
Performing step 4/5, conversion of protected filesets into AFM DR primary filesets.
Successfully completed step 4/5, conversion of protected filesets into AFM DR primary filesets.
Performing step 5/5, creation of output DR configuration file.
Successfully completed step 5/5, creation of output DR configuration file.
```

File to be used with secondary cluster in next step of cluster DR setup: /root//DR_Config

4. Issue the following command on the secondary cluster to create the independent filesets that will later be paired with those on the primary cluster to form AFM DR pairs as part of failing back to a new primary cluster:

```
mmcesdr secondary config --input-file-path /root --prep-outband-transfer
```

The system displays output similar to this:

Creating independent filesets to be used as recipients of AFM DR outbound transfer of data.
 Transfer all data on primary cluster for fileset fs0:combo1 to fileset fs0:combo1 on secondary cluster.
 Transfer all data on primary cluster for fileset fs0:combo2 to fileset fs0:combo2 on secondary cluster.
 Transfer all data on primary cluster for fileset fs0:nfs-ganeshal to fileset fs0:nfs-ganeshal on secondary cluster.
 Transfer all data on primary cluster for fileset fs0:nfs-ganesh2 to fileset fs0:nfs-ganesh2 on secondary cluster.
 Transfer all data on primary cluster for fileset fs0:smb1 to fileset fs0:smb1 on secondary cluster.
 Transfer all data on primary cluster for fileset fs0:smb2 to fileset fs0:smb2 on secondary cluster.
 Transfer all data on primary cluster for fileset fs1:async_dr to fileset fs1:async_dr on secondary cluster.
 Transfer all data on primary cluster for fileset fs1:obj_sofpolicy1 to fileset fs1:obj_sofpolicy1 on secondary cluster.
 mmcesdr: CES Object protocol is not enabled but there is an object related export present.
 Skipping clearing out the object related files and directories from export.
 Transfer all data on primary cluster for fileset fs1:obj_sofpolicy2 to fileset fs1:obj_sofpolicy2 on secondary cluster.
 mmcesdr: CES Object protocol is not enabled but there is an object related export present.
 Skipping clearing out the object related files and directories from export.
 Transfer all data on primary cluster for fileset fs1:object_fileset to fileset fs1:object_fileset on secondary cluster.
 mmcesdr: CES Object protocol is not enabled but there is an object related export present.
 Skipping clearing out the object related files and directories from export.
 Successfully completed creating independent filesets to be used as recipients of AFM DR outbound transfer of data.
 Transfer data from primary cluster through outbound trucking to the newly created independent filesets before proceeding to the next step.

5. After all the data has been transferred to the secondary, issue the following command to complete the set up on the secondary:

```
mmcesdr secondary config --input-file-path /root
```

The system displays output similar to this:

```
Performing step 1/3, verification of independent filesets to be used for AFM DR.
Successfully completed step 1/3, creation of independent filesets to be used for AFM DR.
Successfully completed 1/3, verification of independent filesets to be used for AFM DR.
Performing step 2/3, creation of NFS exports to be used for AFM DR.
Successfully completed step 2/3, creation of NFS exports to be used for AFM DR.
Performing step 3/3, conversion of independent filesets to AFM DR secondary filesets.
Successfully completed step 3/3, conversion of independent filesets to AFM DR secondary filesets.
```

6. Issue the following command on the secondary cluster after the primary cluster has failed:

```
mmcesdr secondary failover
```

The system displays output similar to this:

```
Performing step 1/4, saving current NFS configuration to restore after failback.
Successfully completed step 1/4, saving current NFS configuration to restore after failback.
Performing step 2/4, failover of secondary filesets to primary filesets.
Successfully completed step 2/4, failover of secondary filesets to primary filesets.
Performing step 3/4, protocol configuration/exports restore.
Successfully completed step 3/4, protocol configuration/exports restore.
Performing step 4/4, create/verify NFS AFM DR transport exports.
Successfully completed step 4/4, create/verify NFS AFM DR transport exports.
```

7. Issue the following command on the secondary cluster to prepare recovery snapshots that contain data that will be transferred to the new primary cluster:

```
mmcesdr secondary failback --generate-recovery-snapshots
--output-file-path "/root/" --input-file-path "/root/"
```

The system displays output similar to this:

```
Performing step 1/2, generating recovery snapshots for all AFM DR acting primary filesets.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/combo1/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-1 to fileset link point of fileset fs0:combo1 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/combo2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-2 to fileset link point of fileset fs0:combo2 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/nfs-ganeshal/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-3 to fileset link point of fileset fs0:nfs-ganeshal on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/nfs-ganesh2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-4 to fileset link point of fileset fs0:nfs-ganesh2 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/smb1/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-5 to fileset link point of fileset fs0:smb1 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs0/smb2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-6 to fileset link point of fileset fs0:smb2 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs1/.async_dr/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DECB-2 to fileset link point of fileset fs1:async_dr on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs1/obj_sofpolicy1/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DECB-3 to fileset link point of fileset fs1:obj_sofpolicy1 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs1/obj_sofpolicy2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DECB-4 to fileset link point of fileset fs1:obj_sofpolicy2 on new primary cluster.
Transfer all data under snapshot located on acting primary cluster at:
/gpfs/fs1/object_fileset/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DECB-1 to fileset link point of fileset fs1:object_fileset on new primary cluster.
Successfully completed step 1/2, generating recovery snapshots for all AFM DR acting primary filesets.
Performing step 2/2, creation of recovery output file for failback to new primary.
Successfully completed step 2/2, creation of recovery output file for failback to new primary.
```

File to be used with new primary cluster in next step of failback to new primary cluster: /root//DR_Config

mmcesdr

8. Issue the following command on the primary cluster to restore the protocol and export services configuration information:

```
mmcesdr primary restore --new-primary
```

The system displays output similar to this:

```
Restoring cluster and enabled protocol configurations/exports.  
Successfully completed restoring cluster and enabled protocol configurations/exports.
```

9. Issue the following command on the secondary cluster to restore the protocol and export services configuration information:

```
mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path "/root"
```

The system displays output similar to this:

```
Performing step 1/2, converting protected filesets back into AFM DR secondary filesets.  
Successfully completed step 1/2, converting protected filesets back into AFM DR secondary filesets.  
Performing step 2/2, restoring AFM DR-based NFS share configuration.  
Successfully completed step 2/2, restoring AFM DR-based NFS share configuration.
```

10. Issue the following command on the primary cluster to back up configuration:

```
mmcesdr primary backup
```

The system displays output similar to this:

```
Performing step 1/2, configuration fileset creation/verification.  
Successfully completed step 1/2, configuration fileset creation/verification.  
Performing step 2/2, protocol and export services configuration backup.  
Successfully completed step 2/2, protocol and export services configuration backup.
```

11. Issue the following command on the primary cluster to restore configuration when the primary cluster is not in a protocols DR relationship with another cluster:

```
mmcesdr primary restore --file-config --restore
```

The system displays output similar to this:

```
Restoring cluster and enabled protocol configurations/exports.  
Successfully completed restoring cluster and enabled protocol configurations/exports.
```

```
=====
= If all steps completed successfully, remove and then re-create file
= authentication on the Primary cluster.
= Once this is complete, Protocol Cluster Configuration Restore will be complete.
=====
```

See also

- “mmces command” on page 304
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmsmb command” on page 663
- “mmobj command” on page 581
- “mmuserauth command” on page 690

Location

```
/usr/lpp/mmfs/bin
```

mmchattr command

Changes attributes of one or more GPFS files.

Synopsis

```
mmchattr [-m MetadataReplicas] [-M MaxMetadataReplicas]
          [-r DataReplicas] [-R MaxDataReplicas] [-P DataPoolName]
          [-D {yes | no}] [-I {yes | defer}] [-i {yes | no}]
          [-a {yes | no}] [-l]
          [--set-attr AttributeName[=Value] [--pure-attr-create | --pure-attr-replace] |
          --delete-attr AttributeName [--pure-attr-delete]]]
          [--hex-attr] [--hex-attr-name] [--no-attr-ctime]
          [--compact] [--compression {yes | no}]
          [--block-group-factor BlockGroupFactor]
          [--write-affinity-depth WriteAffinityDepth]
          [--write-affinity-failure-group "WadfgValueString"]
          [--indefinite-retention {yes | no}]
          [--expiration-time yyyy-mm-dd[@hh:mm:ss]]
          Filename [Filename...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchattr** command to change the replication attributes, storage pool assignments, retention and immutability attributes, I/O caching policy, and file compression or decompression of files in the file system.

The replication factor must be less than or equal to the maximum replication factor for the file. If insufficient space is available in the file system to increase the number of replicas to the value requested, the **mmchattr** command ends. However, some blocks of the file might have their replication factor increased after the **mmchattr** command ends. If more free space becomes available in the file system later (when, for example, you add another disk to the file system), you can then issue the **mmrestripefs** command with the **-r** or **-b** option to complete the replication of the file. The **mmrestripefile** command can be used in a similar manner. You can use the **mmisattr** command to display the replication values.

Data of a file is stored in a specific storage pool. A storage pool is a collection of disks or RAID's with similar properties. Because these storage devices have similar properties, you can manage them as a group. You can use storage pools to do the following tasks:

- Partition storage for the file system
- Assign file storage locations
- Improve system performance
- Improve system reliability

The Direct I/O caching policy bypasses file cache and transfers data directly from disk into the user space buffer, as opposed to using the normal cache policy of placing pages in kernel memory. Applications with poor cache hit rates or a large amount of I/O might benefit from the use of Direct I/O.

The **mmchattr** command can be run against a file in use.

You must have write permission for the files whose attributes you are changing.

mmchattr

Parameters

-m *MetadataReplicas*

Specifies how many copies of the file system's metadata to create. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of the *MaxMetadataReplicas* attribute of the file.

-M *MaxMetadataReplicas*

Specifies the maximum number of copies of indirect blocks for a file. Space is reserved in the inode for all possible copies of pointers to indirect blocks. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be less than the value of the *DefaultMetadataReplicas* attribute of the file.

-r *DataReplicas*

Specifies how many copies of the file data to create. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of the *MaxDataReplicas* attribute of the file.

-R *MaxDataReplicas*

Specifies the maximum number of copies of data blocks for a file. Space is reserved in the inode and indirect blocks for all possible copies of pointers to data blocks. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be less than the value of the *DefaultDataReplicas* attribute of the file.

-P *DataPoolName*

Changes the assigned storage pool of the file to the specified *DataPoolName*. The caller must have superuser or root privileges to change the assigned storage pool.

-D {**yes** | **no**}

Enable or disable the Direct I/O caching policy for files.

-I {**yes** | **defer**}

Specifies whether replication and migration between pools, or file compression or decompression, is to be performed immediately (**-I yes**), or deferred until a later call to **mmrestripefs** or **mmrestripefile** (**-I defer**). By deferring the operation, you can complete it when the system is not loaded with processes or I/O. Also, if multiple files are affected, the data movement can be done in parallel. The default is **-I yes**. For more information about file compression and decompression, see the **--compression** option in this topic.

-i {**yes** | **no**}

Specifies whether the file is immutable (**-i yes**) or not immutable (**-i no**).

Note: The immutability attribute is specific to the current instance of the file. Restoring an image of the file to another location does not retain the immutability option. You must set it yourself.

-a {**yes** | **no**}

Specifies whether the file is in **appendOnly** mode (**-a yes**) or not (**-a no**).

Notes:

1. The **appendOnly** setting is specific to the current instance of the file. Restoring an image of the file to another location does not retain the **appendOnly** mode. You must set it yourself.
2. **appendOnly** mode is not supported for AFM filesets.

-l Specifies that this command works only with regular files and directories and does not follow symlinks. The default is to follow symlinks.

--set-attr *AttributeName*[=*Value*]

Sets the specified extended attribute name to the specified *Value* for each file. If no *Value* is specified, **--set-attr *AttributeName*** sets the extended attribute name to a zero-length value.

--pure-attr-create

When this option is used, the command fails if the specified extended attribute exists.

--pure-attr-replace

When this option is used, the command fails if the specified extended attribute does not exist.

--delete-attr *AttributeName*

Removes the extended attribute.

--pure-attr-delete

When this option is used, the command fails if the specified extended attribute does not exist.

--hex-attr

Inputs the attribute value in hex.

--hex-attr-name

Inputs the attribute name in hex.

--no-attr-ctime

Changes the attribute without setting the ctime of the file. This is restricted to root only.

--compact

Converts a directory to GPFS 4.1 format (if needed) and then compacts the directory, potentially reducing its size. If many files were previously removed from the directory, **--compact** can improve the performance of directory operations. The directory name is specified as *Filename*. If the value specified for *Filename* is not a directory, it is ignored.

Note: Directories that are created with a GPFS 4.1 or higher file system, or directories that were previously converted to GPFS 4.1 format, are compacted automatically as files are removed. Using the **--compact** option is not necessary in these instances.

--compression {yes | no}

Compresses or decompresses the specified files. You can use the **-I defer** option to defer the operation until a later call to **mmrestripefs** or **mmrestripefile**. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--block-group-factor *BlockGroupFactor*

Specifies how many file system blocks are laid out sequentially on disk to behave like a single large block. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

--write-affinity-depth *WriteAffinityDepth*

Specifies the allocation policy to be used. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

--write-affinity-failure-group "*WadfgValueString*"

Indicates the range of nodes (in a shared nothing architecture) where replicas of blocks in the file are to be written. You use this parameter to determine the layout of a file in the cluster so as to optimize the typical access patterns of your applications. This applies only to a new data block layout; it does not migrate previously existing data blocks.

"*WadfgValueString*" is a semicolon-separated string identifying one or more failure groups in the following format:

```
FailureGroup1[:FailureGroup2[:FailureGroup3]]
```

where each *FailureGroup_x* is a comma-separated string identifying the rack (or range of racks), location (or range of locations), and node (or range of nodes) of the failure group in the following format:

```
Rack1{:Rack2{:...{:Rackx}}},Location1{:Location2{:...{:Locationx}}},ExtLg1{:ExtLg2{:...{:ExtLgx}}}
```

For example, the following value

```
1,1,1:2;2,1,1:2;2,0,3:4
```

mmchattr

means that the first failure group is on rack 1, location 1, extLg 1 or 2; the second failure group is on rack 2, location 1, extLg 1 or 2; and the third failure group is on rack 2, location 0, extLg 3 or 4.

If the end part of a failure group string is missing, it is interpreted as 0. For example, the following are interpreted the same way:

```
2
2,0
2,0,0
```

Notes:

1. Only the end part of a failure group string can be left off. The missing end part may be the third field only, or it may be both the second and third fields; however, if the third field is provided, the second field must also be provided. The first field must *always* be provided. In other words, every comma must both follow and precede a number; therefore, *none* of the following are valid:

```
2,0,
2,
,0,0
0,,0
,,0
```

2. Wildcard characters (*) are supported in these fields.

--indefinite-retention {yes | no}

Turns indefinite retention on or off. An alternative form of this parameter is **-e {yes | no}**. See **--expiration-time**.

--expiration-time yyyy-mm-dd[@hh:mm:ss]

Specifies the expiration time. An alternative form of this parameter is **-E yyyy-mm-dd[@hh:mm:ss]**. Expiration time and indefinite retention are independent attributes. You can change the value of either one without affecting the value of the other.

Filename

The name of the file to be changed. You must enter at least one file name; if you specify more than one, delimit each file name by a space. Wildcard characters are supported in file names; for example, **project*.sched**.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have write access to the file to run the **mmchattr** command.

You can issue the **mmchattr** command only from a node in the GPFS cluster where the file system is mounted.

Examples

1. To change the metadata replication factor to 2 and the data replication factor to 2 for the **project7.resource** file in file system **fs1**, issue this command:

```
mmchattr -m 2 -r 2 /fs1/project7.resource
```

To confirm the change, issue this command:

```
mmfattr project7.resource
```

The system displays information similar to:

```

replication factors
metadata(max) data(max) file [flags]
-----
2 ( 2) 2 ( 2) /fs1/project7.resource

```

2. Migrating data from one storage pool to another using the **mmchattr** command with the **-I defer** option, or the **mmapplypolicy** command with the **-I defer** option causes the data to be ill-placed. This means that the storage pool assignment for the file has changed, but the file data has not yet been migrated to the assigned storage pool.

The **mmlsattr -L** command causes show ill-placed flags on the files that are ill-placed. The **mmrestripefs**, or **mmrestripefile** command can be used to migrate data to the correct storage pool, and the ill-placed flag is cleared. This is an example of an ill-placed file:

```
mmlsattr -L 16Kfile6.tmp
```

The system displays output similar to this:

```

file name:          16Kfile6.tmp
metadata replication: 1 max 2
data replication:   1 max 2
immutable:         no
appendOnly:       no
flags:            directio
storage pool name: system
fileset name:     root
snapshot name:
creation time:    Thu Mar 28 14:49:23 2013
Misc attributes:  ARCHIVE

```

3. The following example shows the result of using the **--set-attr** parameter.

```

mmchattr --set-attr user.pfs001=testuser 16Kfile7.tmp
mmlsattr -L -d 16Kfile7.tmp

```

The system displays output similar to this:

```

file name:          16Kfile7.tmp
metadata replication: 1 max 2
data replication:   1 max 2
immutable:         no
appendOnly:       no
flags:
storage pool name: system
fileset name:     root
snapshot name:
creation Time:     Fri Feb 24 12:00:13 2012
Misc attributes:  ARCHIVE
user.pfs001:      "testuser"

```

4. To set the write affinity failure group for a file and to see the results, issue these commands:

```

mmchattr --write-affinity-failure-group="64,0,0;128,0,1;128,0,2" /gpfs1/testfile
mmlsattr -L /gpfs1/testfile

```

The system displays output similar to this:

```

file name:          /gpfs1/testfile
metadata replication: 3 max 3
data replication:   3 max 3
immutable:         no
appendOnly:       no
flags:
storage pool name: system
fileset name:     root
snapshot name:
Write Affinity Depth Failure Group(FG) Map for copy:1 64,0,0
Write Affinity Depth Failure Group(FG) Map for copy:2 128,0,1
Write Affinity Depth Failure Group(FG) Map for copy:3 128,0,2
creation time:     Wed Sep 12 02:53:18 2012
Misc attributes:  ARCHIVE

```

mmchattr

See also

- “mmcrfs command” on page 414
- “mmlsattr command” on page 519
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmchcluster command

Changes GPFS cluster configuration data.

Synopsis

```
mmchcluster --ccr-enable
```

or

```
mmchcluster {[--ccr-disable] [-p PrimaryServer] [-s SecondaryServer]}
```

or

```
mmchcluster -p LATEST
```

or

```
mmchcluster {[-r RemoteShellCommand] [-R RemoteFileCopyCommand] [--nouse-sudo-wrapper] |
  --use-sudo-wrapper
```

or

```
mmchcluster -C ClusterName
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmchcluster** command serves several purposes. You can use it to do the following:

1. Change the remote shell and remote file copy programs to be used by the nodes in the cluster.
2. Change the cluster name.
3. Enable or disable the cluster configuration repository (CCR).

When using the traditional server-based (non-CCR) configuration repository, you can also do the following:

1. Change the primary or secondary GPFS cluster configuration server.
2. Synchronize the primary GPFS cluster configuration server.

To display current system information for the cluster, issue the **mmlscluster** command.

For information on how to specify node names, see the topic *Specifying nodes as inputs to GPFS commands* in the *IBM Spectrum Scale: Advanced Administration Guide*.

When issuing the **mmchcluster** command with the **-p** or **-s** options, the specified nodes must be available in order for the command to succeed. If any of the nodes listed are not available when the command is issued, a message listing those nodes is displayed. You must correct the problem on each node and reissue the command.

Attention: The **mmchcluster** command, when issued with either the **-p** or **-s** option, is designed to operate in an environment where the current primary and secondary cluster configuration servers are **not** available. As a result, the command can run without obtaining its regular serialization locks. To assure smooth transition to a new cluster configuration server, no other GPFS commands (**mm** commands) should be running when the command is issued, nor should any other command be issued until the **mmchcluster** command has successfully completed.

mmchcluster

Parameters

--ccr-enable

Enables the configuration server repository (CCR), which stores redundant copies of configuration data files on all quorum nodes. The advantage of CCR over the traditional primary or backup configuration server semantics is that when using CCR, all GPFS administration commands as well as file system mounts and daemon startups work normally as long as a majority of quorum nodes are accessible.

For more information, see the topic *Cluster configuration data files* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

The CCR operation requires the use of the GSKit toolkit for authenticating network connections. As such, the **gpfs.gskit** package, which is available on all Editions, should be installed.

--ccr-disable

Reverts to the traditional primary or backup configuration server semantics and destroys the CCR environment. All nodes must be shut down before disabling CCR.

-p PrimaryServer

Change the primary server node for the GPFS cluster data. This may be specified as a short or long node name, an IP address, or a node number.

LATEST – Synchronize all of the nodes in the GPFS cluster ensuring they are using the most recently specified primary GPFS cluster configuration server. If an invocation of the **mmchcluster** command fails, you are prompted to reissue the command and specify **LATEST** on the **-p** option to synchronize all of the nodes in the GPFS cluster. Synchronization provides for all nodes in the GPFS cluster to use the most recently specified primary GPFS cluster configuration server.

This option only applies when the traditional server-based configuration (non-CCR) repository is used.

-s SecondaryServer

Change the secondary server node for the GPFS cluster data. To remove the secondary GPFS server and continue operating without it, specify a null string, "", as the parameter. This may be specified as a short or long nodename, an IP address, or a node number.

This option only applies when the traditional server-based configuration (non-CCR) repository is used.

-r RemoteShellCommand

Specifies the fully-qualified path name for the remote shell program to be used by GPFS.

The remote shell command must adhere to the same syntax format as the **ssh** command, but may implement an alternate authentication mechanism.

-R RemoteFileCopy

Specifies the fully-qualified path name for the remote file copy program to be used by GPFS.

The remote copy command must adhere to the same syntax format as the **scp** command, but may implement an alternate authentication mechanism.

--nouse-sudo-wrapper

Specifies that the cluster reverts to using the default remote shell program and remote copy program. For more information, see the topic *Running IBM Spectrum Scale without remote root login* in the *IBM Spectrum Scale: Advanced Administration Guide*.

--use-sudo-wrapper

Specifies that the nodes in the cluster call the **ssh** sudo wrapper script and the **scp** sudo wrapper script as the remote shell program and the remote copy program. For more information, see the topic *Running IBM Spectrum Scale without remote root login* in the *IBM Spectrum Scale: Advanced Administration Guide*.

-C ClusterName

Specifies a new name for the cluster. If the user-provided name contains dots, it is assumed to be a fully qualified domain name. Otherwise, to make the cluster name unique, the domain of the first quorum node or, if specified, the primary configuration server will be appended to the user-provided name.

Since each cluster is managed independently, there is no automatic coordination and propagation of changes between clusters like there is between the nodes within a cluster. This means that if you change the name of the cluster, you should notify the administrators of all other GPFS clusters that can mount your file systems so that they can update their own environments.

Before running this option, ensure that all GPFS daemons on all nodes have been stopped.

See the **mmauth**, **mmremotefilesystem**, and **mmremotefs** commands.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchcluster** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To change the primary GPFS server for the cluster, issue this command:

```
mmchcluster -p k164n06
```

The system displays output similar to:

```
mmchcluster: Command successfully completed
```

To confirm the change, issue this command:

```
mmiscluster
```

The system displays information similar to:

```
GPFS cluster information
```

```
=====
```

```
GPFS cluster name:      cluster1.kgn.ibm.com
GPFS cluster id:       680681562214606028
GPFS UID domain:      cluster1.kgn.ibm.com
Remote shell command:  /usr/bin/rsh
Remote file copy command: /usr/bin/rcp
```

```
GPFS cluster configuration servers:
```

```
-----
```

```
Primary server:   k164n06.kgn.ibm.com
Secondary server: k164n05.kgn.ibm.com
```

```
Node Daemon node name  IP address  Admin node name  Designation
```

```
-----
```

```
1 k164n04.kgn.ibm.com 198.117.68.68 k164n04.kgn.ibm.com quorum
2 k164n05.kgn.ibm.com 198.117.68.71 k164n05.kgn.ibm.com quorum
3 k164n06.kgn.ibm.com 198.117.68.70 k164sn06.kgn.ibm.com
```

mmchcluster

See also

- “mmaddnode command” on page 245
- “mmchnode command” on page 381
- “mmcrcluster command” on page 403
- “mmdelnode command” on page 460
- “mmlscluster command” on page 524
- “mmremotecluster command” on page 621

Location

/usr/lpp/mmfs/bin

mmchconfig command

Changes GPFS configuration parameters.

Synopsis

```
mmchconfig Attribute=value[,Attribute=value...] [-i | -I]
           [-N {Node[,Node...] | NodeFile | NodeClass}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchconfig** command to change the GPFS configuration attributes on a single node, a set of nodes, or globally for the entire cluster.

- | If you change both **maxblocksize** and **pagepool** in the same command, follow these rules:
- | • Specify **pagepool** first if you are increasing the values.
- | • Specify **maxblocksize** first if you are decreasing the values.

Results

The configuration is updated on each node in the GPFS cluster.

Parameters

- I Specifies that the changes take effect immediately, but do not persist when GPFS is restarted. This option is valid only for the following attributes:
 - **deadlockBreakupDelay**
 - **deadlockDataCollectionDailyLimit**
 - **deadlockDataCollectionMinInterval**
 - **deadlockDetectionThreshold**
 - **deadlockDetectionThresholdForShortWaiters**
 - | • **deadlockDetectionThresholdIfOverloaded**
 - **deadlockOverloadThreshold**
 - **dmapiEventTimeout**
 - **dmapiMountTimeout**
 - **dmapiSessionFailureTimeout**
 - **expelDataCollectionDailyLimit**
 - **expelDataCollectionMinInterval**
 - **fastestPolicyCmpThreshold**
 - **fastestPolicyMaxValidPeriod**
 - **fastestPolicyMinDiffPercent**
 - **fastestPolicyNumReadSamples**
 - **fileHeatLossPercent**
 - **fileHeatPeriodMinutes**
 - **lrocData**
 - **lrocDataMaxFileSize**
 - **lrocDataStubFileSize**

mmchconfig

- IrocDirectories
 - IrocInodes
 - maxMBpS
 - nsdBufSpace
 - pagepool
 - pitWorkerThreadsPerNode
 - readReplicaPolicy
 - systemLogLevel
 - unmountOnDiskFail
 - verbsRdmaRoCEToS
 - verbsRdmAsPerConnection
 - verbsRdmAsPerNode
 - verbsSendBufferMemoryMB
 - worker1Threads (only when adjusting value down)
- i Specifies that the changes take effect immediately and are permanent. This option is valid only for the following attributes:
- cesSharedRoot
 - cnfsGrace
 - cnfsMountdPort
 - cnfsNFSDprocs
 - cnfsReboot
 - cnfsSharedRoot
 - cnfsVersions
 - dataDiskWaitTimeForRecovery
 - deadlockBreakupDelay
 - deadlockDataCollectionDailyLimit
 - deadlockDataCollectionMinInterval
 - deadlockDetectionThreshold
 - deadlockDetectionThresholdForShortWaiters
 - deadlockDetectionThresholdIfOverloaded
 - deadlockOverloadThreshold
 - disableInodeUpdateOnFdatasync
 - dmapiEventTimeout
 - dmapiMountTimeoout
 - dmapiSessionFailureTimeout
 - expelDataCollectionDailyLimit
 - expelDataCollectionMinInterval
 - fastestPolicyCmpThreshold
 - fastestPolicyMaxValidPeriod
 - fastestPolicyMinDiffPercent
 - fastestPolicyNumReadSamples
 - fileHeatLossPercent
 - fileHeatPeriodMinutes
 - forceLogWriteOnFdatasync
 - IrocData

- `IrocDataMaxFileSize`
- `IrocDataStubFileSize`
- `IrocDirectories`
- `IrocInodes`
- `maxDownDisksForRecovery`
- `maxFailedNodesForRecovery`
- `maxMBpS`
- `metadataDiskWaitTimeForRecovery`
- `minDiskWaitTimeForRecovery`
- `nsdBufSpace`
- `pagepool`
- `pitWorkerThreadsPerNode`
- `readReplicaPolicy`
- `restripeOnDiskFailure`
- `systemLogLevel`
- `unmountOnDiskFail`
- `verbsRdmaRoCEToS`
- `verbsRdmAsPerConnection`
- `verbsRdmAsPerNode`
- `verbsSendBufferMemoryMB`
- `worker1Threads` (only when adjusting value down)

-N {*Node* [,*Node* ...] | *NodeFile* | *NodeClass*}

Specifies the set of nodes to which the configuration changes apply. The default is **-N all**.

For information on how to specify node names, see the topic *Specifying nodes as inputs to GPFS commands* in the *IBM Spectrum Scale: Advanced Administration Guide*.

To see a complete list of the attributes for which the **-N** flag is valid, see the list of node names allowed in the topic *Changing the GPFS cluster configuration data* in the *IBM Spectrum Scale: Advanced Administration Guide*.

This command does not support a *NodeClass* of **mount**.

Attribute=value

Specifies the name of the attribute to be changed and its associated *value*. More than one attribute and value pair can be specified. To restore the GPFS default setting for an attribute, specify **DEFAULT** as its *value*.

This command accepts the following attributes:

adminMode

Specifies whether all nodes in the cluster are used for issuing GPFS administration commands or just a subset of the nodes. Valid values are:

a11ToA11

Indicates that all nodes in the cluster are used for running GPFS administration commands and that all nodes are able to execute remote commands on any other node in the cluster without the need of a password.

central

Indicates that only a subset of the nodes is used for running GPFS commands and that only those nodes are able to execute remote commands on the rest of the nodes in the cluster without the need of a password.

mmchconfig

For additional information, see the following *IBM Spectrum Scale: Administration and Programming Reference* topic: "Requirements for administering a GPFS file system" on page 1.

afmAsyncDelay

Specifies (in seconds) the amount of time by which write operations are delayed (because write operations are asynchronous with respect to remote clusters). For write-intensive applications that keep writing to the same set of files, this delay is helpful because it replaces multiple writes to the home cluster with a single write containing the latest data. However, setting a very high value weakens the consistency of data on the remote cluster.

This configuration parameter is applicable only for writer caches (SW and IW), where data from cache is pushed to home.

Valid values are 1 through 2147483647. The default is 15.

afmDirLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a directory, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of that directory has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 60. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmDirOpenRefreshInterval

Controls the frequency of data revalidations that are triggered by such I/O operations as **read** or **write** (specified in seconds). After a directory has been cached, **open** requests resulting from I/O operations on that object are directed to the cached directory until the specified amount of time has passed. Once the specified amount of time has passed, the **open** request gets directed to a gateway node rather than to the cached directory.

Valid values are 0 through 2147483647. The default is 60. Setting a lower value guarantees a higher level of consistency.

afmDisconnectTimeout

The Waiting period in seconds to detect the status of the home cluster. If the home cluster is inaccessible, the metadata server (MDS) changes the state from 'cache' to 'disconnected'.

afmExpirationTimeout

Is used with **afmDisconnectTimeout** (which can be set only through **mmchconfig**) to control how long a network outage between the cache and home clusters can continue before the data in the cache is considered out of sync with home. After **afmDisconnectTimeout** expires, cached data remains available until **afmExpirationTimeout** expires, at which point the cached data is considered expired and cannot be read until a reconnect occurs.

Valid values are 0 through 2147483647. The default is 300.

afmFileLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a file, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of the file has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 30. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmFileOpenRefreshInterval

Controls the frequency of data revalidations that are triggered by such I/O operations as **read** or **write** (specified in seconds). After a file has been cached, **open** requests resulting from I/O

operations on that object are directed to the cached file until the specified amount of time has passed. Once the specified amount of time has passed, the **open** request gets directed to a gateway node rather than to the cached file.

Valid values are 0 through 2147483647. The default is 30. Setting a lower value guarantees a higher level of consistency.

afmHardMemThreshold

Sets a limit to the maximum amount of memory that AFM can use on each gateway node to record changes to the file system. After this limit is reached, the fileset goes into a 'dropped' state.

Exceeding the limit and the fileset going into a 'dropped' state due to accumulated pending requests might occur if -

- the cache cluster is disconnected for an extended period of time.
- the connection with the home cluster is on a low bandwidth.

Reboot the gateway node after you change the value.

afmHashVersion

Specifies an older or newer version of gateway node hashing algorithm (for example, **mmchconfig afmHashVersion=2**). This can be used to minimize the impact of gateway nodes joining or leaving the active cluster by running as few recoveries as much as possible. Valid values are 1 or 2.

afmNumReadThreads

Defines the number of threads that can be used on each participating gateway node during parallel read. The default value of this parameter is 1; that is, one reader thread will be active on every gateway node for each big write operation qualifying for splitting per the parallel read threshold value. The valid range of values is 1 to 64.

afmNumWriteThreads

Defines the number of threads that can be used on each participating gateway node during parallel write. The default value of this parameter is 1; that is, one writer thread will be active on every gateway node for each big write operation qualifying for splitting per the parallel write threshold value. Valid values can range from 1 to 64.

afmParallelReadChunkSize

Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel reads. Values are interpreted in terms of bytes. The default value of this parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelReadThreshold

Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default value is 1024 MB. The valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelWriteChunkSize

Defines the minimum chunk size of the write that needs to be distributed among the gateway nodes during parallel writes. Values are interpreted in terms of bytes. The default value of this parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelWriteThreshold

Defines the threshold beyond which parallel writes become effective. Writes are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default

mmchconfig

value of this parameter is 1024 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmReadSparseThreshold

Specifies the size in MB for files in cache beyond which sparseness is maintained. For all files below the specified threshold, sparseness is not maintained.

afmSecondaryRW

Specifies if the secondary is read-write or not.

yes

Specifies that the secondary is read-write.

no

Specifies that the secondary is not read-write.

afmShowHomeSnapshot

Controls the visibility of the home snapshot directory in cache. For this to be visible in cache, this variable has to be set to **yes**, and the snapshot directory name in cache and home should not be the same.

yes

Specifies that the home snapshot link directory is visible.

no

Specifies that the home snapshot link directory is not visible.

See also the topic about peer snapshots in the *IBM Spectrum Scale: Advanced Administration Guide*.

atimeDeferredSeconds

Controls the update behavior of **atime** when the **relatime** option is enabled. The default value is 86400 seconds (24 hours). A value of 0 effectively disables **relatime** and causes the behavior to be the same as the **atime** setting.

For more information, see the topic *GPFS-specific mount options* in the *IBM Spectrum Scale: Advanced Administration Guide*.

autoload

Starts GPFS automatically whenever the nodes are rebooted. Valid values are **yes** or **no**.

The **-N** flag is valid for this attribute.

automountDir

Specifies the directory to be used by the Linux automounter for GPFS file systems that are being mounted automatically. The default directory is `/gpfs/automountdir`. This parameter does not apply to AIX and Windows environments.

cesSharedRoot

Specifies a directory in a GPFS file system to be used by the Cluster Export Services (CES) subsystem.

GPFS must be down on all CES nodes in the cluster when changing the **cesSharedRoot** attribute.

cipherList

Sets the security mode for the cluster. The security mode determines the level of the security that the cluster provides for communications between nodes in the cluster and also for communications with other clusters. There are three security modes:

EMPTY

The sending node and the receiving node do not authenticate each other, do not encrypt transmitted data, and do not check data integrity.

AUTHONLY

The sending and receiving nodes authenticate each other, but they do not encrypt transmitted data and do not check data integrity. This mode is the default in IBM Spectrum Scale V4.2 or later.

Cipher

The sending and receiving nodes authenticate each other, encrypt transmitted data, and check data integrity. To set this mode, you must specify the name of a supported cipher, such as AES128-GCM-SHA256.

For more information about the security mode and supported ciphers, see the topic *Security mode* in the *IBM Spectrum Scale: Advanced Administration Guide*.

cnfsGrace

Specifies the number of seconds a CNFS node will deny new client requests after a node failover or failback, to allow clients with existing locks to reclaim them without the possibility of some other client being granted a conflicting access. For v3, only new lock requests are denied. For v4, new lock, read and write requests are rejected. Note that the **cnfsGrace** value also determines the time period for the server lease.

Valid values are 10 through 600. The default is 90 seconds. A short grace period is good for fast server failover, however it comes at the cost of increased load on server to effect lease renewal.

GPFS must be down on all CNFS nodes in the cluster when changing the **cnfsGrace** attribute.

cnfsMountdPort

Specifies the port number to be used for **rpc.mountd**. See the *IBM Spectrum Scale: Advanced Administration Guide* for restrictions and additional information.

cnfsNFSDprocs

Specifies the number of **nfsd** kernel threads. The default is 32.

cnfsReboot

Specifies whether the node will reboot when CNFS monitoring detects an unrecoverable problem that can only be handled by node failover.

Valid values are **yes** or **no**. The default is **yes** and recommended. If node reboot is not desired for other reasons, it should be noted that clients that were communicating with the failing node are likely to get errors or hang. CNFS failover is only guaranteed with **cnfsReboot** enabled.

The **-N** flag is valid for this attribute.

cnfsSharedRoot

Specifies a directory in a GPFS file system to be used by the clustered NFS subsystem.

GPFS must be down on all CNFS nodes in the cluster when changing the **cnfsSharedRoot** attribute.

See the *IBM Spectrum Scale: Advanced Administration Guide* for restrictions and additional information.

cnfsVersions

Specifies a comma-separated list of protocol versions that CNFS should start and monitor.

The default is 3,4.

GPFS must be down on all CNFS nodes in the cluster when changing the **cnfsVersions** attribute.

See the *IBM Spectrum Scale: Advanced Administration Guide* for additional information.

dataDiskWaitTimeForRecovery

Specifies a period of time, in seconds, during which the recovery of **dataOnly** disks is suspended to give the disk subsystem a chance to correct itself. This parameter is taken into account when

mmchconfig

the affected disks belong to a single failure group. If more than one failure group is affected, the delay is based on the value of **minDiskWaitTimeForRecovery**.

Valid values are between 0 and 3600 seconds. The default is 3600. If **restripeOnDiskFailure** is **no**, **dataDiskWaitTimeForRecovery** has no effect.

deadlockBreakupDelay

Specifies how long to wait after a deadlock is detected before attempting to break up the deadlock. Enough time must be provided to allow the debug data collection to complete.

The default is 0, which means that the automated deadlock breakup is disabled. A positive value will enable the automated deadlock breakup. If automated deadlock breakup is to be enabled, a delay of 300 seconds or longer is recommended.

deadlockDataCollectionDailyLimit

Specifies the maximum number of times that debug data can be collected every 24 hours.

The default is 10. If the value is 0, then no debug data is collected when a potential deadlock is detected.

deadlockDataCollectionMinInterval

Specifies the minimum interval between two consecutive collections of debug data.

The default is 300 seconds.

deadlockDetectionThreshold

Specifies the deadlock detection threshold. A suspected deadlock is detected when a waiter waits longer than **deadlockDetectionThreshold**.

The default is 300 seconds. If the value is 0, then automated deadlock detection is disabled.

deadlockDetectionThresholdForShortWaiters

Specifies the deadlock detection threshold for short waiters that should never be long.

The default is 60 seconds.

deadlockDetectionThresholdIfOverloaded

Specifies the deadlock detection threshold to use when a cluster is overloaded.

The default is 1800 seconds.

deadlockOverloadThreshold

Specifies the threshold for detecting a cluster overload condition.

The default is 5 seconds. If the value is 0, then overload detection is disabled.

defaultHelperNodes

For commands that distribute work among a set of nodes, the **defaultHelperNodes** parameter specifies the nodes to be used. When specifying values, follow the rules described for the **-N** parameter.

To override this setting when using such commands, explicitly specify the helper nodes with **-N**.

The commands that use **-N** for this purpose are the following: **mmadddisk**, **mmapplypolicy**, **mmbackup**, **mmchdisk**, **mmcheckquota**, **mmdefragfs**, **mmdeldisk**, **mmdelsnapshot**, **mmfileid**, **mmfsck**, **mmimgbackup**, **mmimgrestore**, **mmrestorefs**, **mmrestripefs**, and **mmrpldisk**.

NodeClass values are listed.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

defaultMountDir

Specifies the default parent directory for GPFS file systems. The default value is **/gpfs**. If an explicit mount directory is not provided with the **mmcrfs**, **mmchfs**, or **mmremotefs** command, the default mount point is set to *DefaultMountDir/DeviceName*.

disableInodeUpdateOnFdatasync

Controls the inode update on `fdatasync` for `mtime` and `atime` updates. Valid values are **yes** or **no**

When **disableInodeUpdateOnFdatasync** is set to **yes**, the inode object is not updated on disk for `mtime` and `atime` updates on `fdatasync()` calls. File size updates are always synced to the disk.

When **disableInodeUpdateOnFdatasync** is set to **no**, the inode object is updated with the current `mtime` on `fdatasync()` calls. This is the default.

dmapiDataEventRetry

Controls how GPFS handles data events that are enabled again immediately after the event is handled by the DMAPI application. Valid values are as follows:

- 1 Specifies that GPFS always regenerates the event as long as it is enabled. This value should only be used when the DMAPI application recalls and migrates the same file in parallel by many processes at the same time.
- 0 Specifies to never regenerate the event. This value should not be used if a file could be migrated and recalled at the same time.

RetryCount

Specifies the number of times the data event should be retried. The default is 2.

For further information regarding DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

dmapiEventTimeout

Controls the blocking of file operation threads of NFS, while in the kernel waiting for the handling of a DMAPI synchronous event. The parameter value is the maximum time, in milliseconds, the thread blocks. When this time expires, the file operation returns **ENOTREADY**, and the event continues asynchronously. The NFS server is expected to repeatedly retry the operation, which eventually finds the response of the original event and continue. This mechanism applies only to read, write, and truncate event types, and only when such events come from NFS server threads. The timeout value is given in milliseconds. The value 0 indicates immediate timeout (fully asynchronous event). A value greater than or equal to 86400000 (which is 24 hours) is considered *infinity* (no timeout, fully synchronous event). The default value is 86400000.

For further information regarding DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

The **-N** flag is valid for this attribute.

dmapiMountEvent

Controls the generation of the **mount**, **preunmount**, and **unmount** events. Valid values are:

all

mount, **preunmount**, and **unmount** events are generated on each node. This is the default behavior.

SessionNode

mount, **preunmount**, and **unmount** events are generated on each node and are delivered to the session node, but the session node does not deliver the event to the DMAPI application unless the event is originated from the **SessionNode** itself.

LocalNode

mount, **preunmount**, and **unmount** events are generated only if the node is a session node.

The **-N** flag is valid for this attribute.

For further information regarding DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

mmchconfig

dmapiMountTimeout

Controls the blocking of **mount** operations, waiting for a disposition for the mount event to be set. This timeout is activated, at most once on each node, by the first external mount of a file system that has DMAPI enabled, and only if there has never before been a mount disposition. Any **mount** operation on this node that starts while the timeout period is active waits for the mount disposition. The parameter value is the maximum time, in seconds, that the **mount** operation waits for a disposition. When this time expires and there is still no disposition for the mount event, the **mount** operation fails, returning the **EIO** error. The timeout value is given in full seconds. The value 0 indicates immediate timeout (immediate failure of the mount operation). A value greater than or equal to 86400 (which is 24 hours) is considered *infinity* (no timeout, indefinite blocking until there is a disposition). The default value is 60.

The **-N** flag is valid for this attribute.

For further information regarding DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

dmapiSessionFailureTimeout

Controls the blocking of file operation threads, while in the kernel, waiting for the handling of a DMAPI synchronous event that is enqueued on a session that has experienced a failure. The parameter value is the maximum time, in seconds, the thread waits for the recovery of the failed session. When this time expires and the session has not yet recovered, the event is cancelled and the file operation fails, returning the **EIO** error. The timeout value is given in full seconds. The value 0 indicates immediate timeout (immediate failure of the file operation). A value greater than or equal to 86400 (which is 24 hours) is considered *infinity* (no timeout, indefinite blocking until the session recovers). The default value is 0.

For further information regarding DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

The **-N** flag is valid for this attribute.

enableIPv6

Controls whether the GPFS daemon communicates through the IPv6 network. The following values are valid:

no Specifies that the GPFS daemon does not communicate through the IPv6 network. This is the default.

yes

Specifies that the GPFS daemon communicates through the IPv6 network. **yes** requires that the daemon be down on all nodes.

prepare

After the command completes, the daemons can be recycled on all nodes at a time chosen by the user (before proceeding to run the command with **commit** specified).

commit

Verifies that all currently active daemons have received the new value, allowing the user to add IPv6 nodes to the cluster.

Note: Before changing the value of **enableIPv6**, the GPFS daemon on the primary configuration server must be inactive. After changing the parameter, the GPFS daemon on the rest of nodes within the cluster should be recycled. This can be done one node a time.

To use IPv6 addresses for GPFS, the operating system must be properly configured as IPv6 enabled, and IPv6 addresses must be configured on all the nodes within the cluster.

enforceFilesetQuotaOnRoot

Controls whether fileset quotas should be enforced for the root user the same way as for any other users. Valid values are **yes** or **no**. The default is **no**.

expelDataCollectionDailyLimit

Specifies the maximum number of times that debug data associated with expelling nodes can be collected in a 24-hour period. Sometimes exceptions are made to help capture the most relevant debug data.

The default is 10. If the value is 0, then no expel-related debug data is collected.

expelDataCollectionMinInterval

Specifies the minimum interval, in seconds, between two consecutive expel-related data collection attempts on the same node.

The default is 120 seconds.

failureDetectionTime

Indicates to GPFS the amount of time it takes to detect that a node has failed.

GPFS must be down on all the nodes when changing the **failureDetectionTime** attribute.

fastestPolicyCmpThreshold

Indicates the disk comparison count threshold, above which GPFS forces selection of this disk as the preferred disk to read and update its current speed.

Valid values are ≥ 3 . The default is 50.

fastestPolicyMaxValidPeriod

Indicates the time period after which the disk's current evaluation is considered invalid (even if its comparison count has exceeded the threshold) and GPFS prefers to read this disk in the next selection to update its latest speed evaluation.

Valid values are ≥ 1 in seconds. The default is 600 (10 minutes).

fastestPolicyMinDiffPercent

A percentage value indicating how GPFS selects the fastest between two disks. For example, if you use the default **fastestPolicyMinDiffPercent** value of 50, GPFS selects a disk as faster only if it is 50% faster than the other. Otherwise, the disks remain in the existing read order.

Valid values are 0 through 100 in percentage points. The default is 50.

fastestPolicyNumReadSamples

Controls how many read samples are taken to evaluate the disk's recent speed.

Valid values are 3 through 100. The default is 5.

fileHeatLossPercent

Specifies the reduction rate of **FILE_HEAT** value for every **fileHeatPeriodMinutes** of file inactivity. The default value is 10.

fileHeatPeriodMinutes

Specifies the inactivity time before a file starts to lose **FILE_HEAT** value. The default value is 0, which means that **FILE_HEAT** is not tracked.

FIPS1402mode

Controls whether GPFS operates in FIPS 140-2 mode, which requires using a FIPS-compliant encryption module for all encryption and decryption activity. Valid values are **yes** or **no**. The default value is **no**.

For FIPS 140-2 considerations, consult the "Encryption" topic in the *IBM Spectrum Scale: Advanced Administration Guide*.

forceLogWriteOnFdatasync

Controls forcing log writes to disk. Valid values are **yes** or **no**.

When **forceLogWriteOnFdatasync** is set to **yes**, the GPFS log record is flushed to disk every time **fdatasync()** is invoked. This is the default.

mmchconfig

When **forceLogWriteOnFdatasync** is set to **no**, the GPFS log record is flushed only when a new block is written to the file.

IrocData

Controls whether user data is populated into the local read-only cache. Other configuration options can be used to select the data that is eligible for the local read-only cache. When using more than one such configuration option, data that matches any of the specified criteria is eligible to be saved.

Valid values are **yes** or **no**. The default value is **yes**.

If **IrocData** is set to **yes**, by default the data that was not already in the cache when accessed by a user is subsequently saved to the local read-only cache. The default behavior can be overridden using the **IrocDataMaxFileSize** and **IrocDataStubFileSize** configuration options to save all data from small files or all data from the initial portion of large files.

IrocDataMaxFileSize

Limits the data that may be saved in the local read-only cache to only the data from small files.

A value of -1 indicates that all data is eligible to be saved. A value of 0 indicates that small files are not to be saved. A positive value indicates the maximum size of a file to be considered for the local read-only cache. For example, a value of 32768 indicates that files with 32 KB of data or less are eligible to be saved in the local read-only cache. The default value is 0.

IrocDataStubFileSize

Limits the data that may be saved in the local read-only cache to only the data from the first portion of all files.

A value of -1 indicates that all file data is eligible to be saved. A value of 0 indicates that stub data is not eligible to be saved. A positive value indicates that the initial portion of each file that is eligible is to be saved. For example, a value of 32768 indicates that the first 32 KB of data from each file is eligible to be saved in the local read-only cache. The default value is 0.

IrocDirectories

Controls whether directory blocks is populated into the local read-only cache. The option also controls other file system metadata such as indirect blocks, symbolic links, and extended attribute overflow blocks.

Valid values are **yes** or **no**. The default value is **yes**.

IrocInodes

Controls whether inodes from open files is populated into the local read-only cache; the cache contains the full inode, including all disk pointers, extended attributes, and data.

Valid values are **yes** or **no**. The default value is **yes**.

maxblocksize

Changes the maximum file system block size. Valid block sizes are 64 KiB, 128 KiB, 256 KiB, 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, and 16 MiB. The default maximum block size is 1 MiB. Specify this value with the character **K** or **M**; for example, use **2M** to specify a block size of 2 MiB.

File systems with block sizes larger than the specified value cannot be created or mounted unless the block size is increased.

GPFS must be down on all the nodes in the cluster when changing the **maxblocksize** attribute.

The **-N** flag is valid for this attribute.

maxDownDisksForRecovery

Specifies the maximum number of disks that may experience a failure and still be subject to an automatic recovery attempt. If this value is exceeded, no automatic recovery actions take place.

Valid values are between 0 and 300. The default is 16. If **restripeOnDiskFailure** is **no**, **maxDownDisksForRecovery** has no effect.

maxFailedNodesForRecovery

Specifies the maximum number of nodes that may be unavailable before automatic disk recovery actions are cancelled.

Valid values are between 0 and 300. The default is 3. If **restripeOnDiskFailure** is **no**, **maxFailedNodesForRecovery** has no effect.

maxFcntlRangesPerFile

Specifies the number of **fcntl** locks that are allowed per file. The default is 200. The minimum value is 10 and the maximum value is 200000.

maxFilesToCache

Specifies the number of inodes to cache for recently used files that have been closed.

Storing the inode of a file in cache permits faster re-access to the file. The default is 4000, but increasing this number may improve throughput for workloads with high file reuse. However, increasing this number excessively may cause paging at the file system manager node. The value should be large enough to handle the number of concurrently open files plus allow caching of recently used files.

The **-N** flag is valid for this attribute.

maxMBpS

Specifies an estimate of how many megabytes of data can be transferred per second into or out of a single node. The default is 2048 MB per second. The value is used in calculating the amount of I/O that can be done to effectively prefetch data for readers and write-behind data from writers. By lowering this value, you can artificially limit how much I/O one node can put on all of the disk servers.

The **-N** flag is valid for this attribute.

maxStatCache

Specifies the number of inodes to keep in the stat cache. The stat cache maintains only enough inode information to perform a query on the file system. If the user did not specify values for **maxFilesToCache** and **maxStatCache**, the default value of **maxFilesToCache** is 4000 and the default value of **maxStatCache** is 1000. However, if the user specified a value for **maxFilesToCache** but not for **maxStatCache**, the default value of **maxStatCache** changes to $4 * \text{maxFilesToCache}$.

The **-N** flag is valid for this attribute.

Note: The stat cache is not effective on the Linux platform. Therefore, you need to set the **maxStatCache** attribute to a smaller value, such as 512, on that platform.

metadataDiskWaitTimeForRecovery

Specifies a period of time, in seconds, during which the recovery of metadata disks is suspended to give the disk subsystem a chance to correct itself. This parameter is taken into account when the affected disks belong to a single failure group. If more than one failure group is affected, the delay is based on the value of **minDiskWaitTimeForRecovery**.

Valid values are between 0 and 3600 seconds. The default is 2400. If **restripeOnDiskFailure** is **no**, **metadataDiskWaitTimeForRecovery** has no effect.

minDiskWaitTimeForRecovery

Specifies a period of time, in seconds, during which the recovery of disks is suspended to give the disk subsystem a chance to correct itself. This parameter is taken into account when more than one failure group is affected. If the affected disks belong to a single failure group, the delay is based on the values of **dataDiskWaitTimeForRecovery** and **metadataDiskWaitTimeForRecovery**.

Valid values are between 0 and 3600 seconds. The default is 1800. If **restripeOnDiskFailure** is **no**, **minDiskWaitTimeForRecovery** has no effect.

mmchconfig

mmapRangeLock

Specifies POSIX or non-POSIX **mmap** byte-range semantics. Valid values are **yes** or **no** (**yes** is the default). A value of **yes** indicates POSIX byte-range semantics apply to **mmap** operations. A value of **no** indicates non-POSIX **mmap** byte-range semantics apply to **mmap** operations.

If using InterProcedural Analysis (IPA), turn this option off:

```
mmchconfig mmapRangeLock=no -i
```

This will allow more lenient intranode locking, but impose internode whole file range tokens on files using **mmap** while writing.

nistCompliance

Controls whether GPFS operates in the NIST 800-131A mode. (This applies to security transport only, not to encryption, as encryption always uses NIST-compliant mechanisms.)

Valid values are:

off

Specifies that there is no compliance to NIST standards. For clusters operating below the GPFS 4.1 level, this is the default.

SP800-131A

Specifies that security transport is to follow the NIST SP800-131A recommendations. For clusters at the GPFS 4.1 level or higher, this is the default.

Note: In a remote cluster setup, all clusters must have the same **nistCompliance** value.

noSpaceEventInterval

Specifies the time interval between calling a callback script of two **noDiskSpace** events of a file system. The default value is 120 seconds. If this value is set to zero, the **noDiskSpace** event is generated every time the file system encounters the **noDiskSpace** event. The **noDiskSpace** event is generated when a callback script is registered for this event with the **mmaddcallback** command.

nsdBufSpace

This option specifies the percentage of the page pool reserved for the network transfer of NSD requests. Valid values are within the range of 10 to 70. The default value is 30. On GPFS Native RAID recovery group NSD servers, this value should be decreased to its minimum of 10, since vdisk-based NSDs are served directly from the RAID buffer pool (as governed by **nsdRAIDBufferPoolSizePct**). On all other NSD servers, increasing either this value or the amount of page pool, or both, could improve NSD server performance. On NSD client-only nodes, this parameter is ignored. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

The **-N** flag is valid for this attribute.

nsdRAIDTracks

This option specifies the number of tracks in the GPFS Native RAID buffer pool, or 0 if this node does not have a GPFS Native RAID vdisk buffer pool. This controls whether GPFS Native RAID services are configured. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

Valid values are: 0; 256 or greater.

The **-N** flag is valid for this attribute.

nsdRAIDBufferPoolSizePct

This option specifies the percentage of the page pool that is used for the GPFS Native RAID vdisk buffer pool. Valid values are within the range of 10 to 90. The default is 50 when GPFS Native RAID is configured on the node in question; 0 when it is not. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

The **-N** flag is valid for this attribute.

nsdServerWaitTimeForMount

When mounting a file system whose disks depend on NSD servers, this option specifies the number of seconds to wait for those servers to come up. The decision to wait is controlled by the criteria managed by the **nsdServerWaitTimeWindowOnMount** option.

Valid values are between 0 and 1200 seconds. The default is 300. A value of zero indicates that no waiting is done. The interval for checking is 10 seconds. If **nsdServerWaitTimeForMount** is 0, **nsdServerWaitTimeWindowOnMount** has no effect.

The mount thread waits when the daemon delays for safe recovery. The mount wait for NSD servers to come up, which is covered by this option, occurs after expiration of the recovery wait allows the mount thread to proceed.

The **-N** flag is valid for this attribute.

nsdServerWaitTimeWindowOnMount

Specifies a window of time (in seconds) during which a mount can wait for NSD servers as described for the **nsdServerWaitTimeForMount** option. The window begins when quorum is established (at cluster startup or subsequently), or at the last known failure times of the NSD servers required to perform the mount.

Valid values are between 1 and 1200 seconds. The default is 600. If **nsdServerWaitTimeForMount** is 0, **nsdServerWaitTimeWindowOnMount** has no effect.

The **-N** flag is valid for this attribute.

When a node rejoins the cluster after having been removed for any reason, the node resets all the failure time values that it knows about. Therefore, when a node rejoins the cluster it believes that the NSD servers have not failed. From the perspective of a node, old failures are no longer relevant.

GPFS checks the cluster formation criteria first. If that check falls outside the window, GPFS then checks for NSD server fail times being within the window.

numaMemoryInterleave

In a Linux NUMA environment, the default memory policy is to allocate memory from the local NUMA node of the CPU from which the allocation request was made. This parameter is used to change to an interleave memory policy for GPFS by starting GPFS with **numactl --interleave=all**. This parameter should be used when the GPFS memory usage needs to be balanced across all NUMA nodes, such as the case when the size of the GPFS page pool exceeds the size of any one NUMA node.

Valid values are **yes** and **no**. The default is **no**.

Before using this parameter, ensure that the Linux `numactl` package has been installed.

pagepool

Changes the size of the cache on each node. The default value is either one-third of the physical memory on the node or 1G, whichever is smaller. This applies to new installations only; on upgrades the existing default value is kept.

The maximum GPFS page pool size depends on the value of the **pagepoolMaxPhysMemPct** parameter and the amount of physical memory on the node. You can specify this value with the suffix **K**, **M**, or **G**, for example, **128M**.

The **-N** flag is valid for this attribute.

mmchconfig

pagepoolMaxPhysMemPct

Percentage of physical memory that can be assigned to the page pool. Valid values are 10 through 90 percent. The default is 75 percent (with the exception of Windows, where the default is 50 percent).

The **-N** flag is valid for this attribute.

pitWorkerThreadsPerNode

Controls the maximum number of threads to be involved in parallel processing on each node that is serving as a Parallel Inode Traversal (PIT) worker.

By default, when a command that uses the PIT engine is run, the file system manager asks all nodes in the local cluster to serve as PIT workers; however, you can specify an exact set of nodes to serve as PIT workers by using the **-N** option of a PIT command. Note that the current file system manager node is a mandatory participant, even if it is not in the list of nodes you specify. On each participating node, up to **pitWorkerThreadsPerNode** can be involved in parallel processing. The range of accepted values is 0 to 8192. The default value varies within the 2-16 range, depending on the file system configuration.

prefetchThreads

Controls the maximum possible number of threads dedicated to prefetching data for files that are read sequentially, or to handle sequential write-behind.

Functions in the GPFS daemon dynamically determine the actual degree of parallelism for prefetching data. The default value is 72. The minimum value is 2. The maximum value of **prefetchThreads** plus **worker1Threads** plus **nsdMaxWorkerThreads** is 8192 on all 64-bit platforms.

The **-N** flag is valid for this attribute.

profile

Specifies a predefined profile of attributes to be applied. System-defined profiles are located in `/usr/lpp/mmfs/profiles/`. All the configuration attributes listed under a cluster stanza are changed as a result of this command. The following system-defined profile names are accepted:

- **gpfsProtocolDefaults**
- **gpfsProtocolRandomIO**

A user's profiles must be installed in `/var/mmfs/etc/`. The profile file specifies GPFS configuration parameters with values different than the documented defaults. A user-defined profile must not begin with the string 'gpfs' and must have the .profile suffix.

User-defined profiles consist of the following stanzas:

```
%cluster:  
[CommaSeparatedNodesOrNodeClasses:]ClusterConfigurationAttribute=Value  
...
```

File system attributes and values are ignored.

A sample file can be found in `/usr/lpp/mmfs/samples/sample.profile`. See the **mmchconfig** command for a detailed description of the different configuration parameters. User-defined profiles should be used only by experienced administrators. When in doubt, use the **mmchconfig** command instead.

readReplicaPolicy

Specifies the location from which the FPO policy is to read replicas. By default, the FPO policy reads the first replica whether there is a replica on the local disk or not. When **readReplicaPolicy=local** is specified, the policy reads replicas from the local disk if the local disk has data; for performance considerations, this is the recommended setting for FPO environments. When **readReplicaPolicy=fastest** is specified, the policy reads replicas from the disk considered the fastest based on the read I/O statistics of the disk. You can tune the way the system determines the fastest policy using the following parameters:

- **fastestPolicyNumReadSamples**
- **fastestPolicyCmpThreshold**
- **fastestPolicyMaxValidPeriod**
- **fastestPolicyMinDiffPercent**

To return this attribute to the default setting, specify **readReplicaPolicy=DEFAULT -i**.

release=LATEST

Changes the IBM Spectrum Scale configuration information to the latest format that is supported by the currently installed level of the product. Perform this operation after you have migrated all the nodes in the cluster to the latest level of the product. For more information, see the topic *Completing the migration to a new level of GPFS in the IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

The command tries to access each node in the cluster to verify the level of the installed code. If the command cannot reach one or more nodes, you must rerun the command until it verifies the information for all the nodes.

restripeOnDiskFailure

Specifies whether GPFS will attempt to automatically recover from certain common disk failure situations.

When a disk experiences a failure and becomes unavailable, the recovery procedure will first attempt to restart the disk and if this fails, the disk is suspended and its data moved to other disks. Similarly, when a node joins the cluster, all disks for which the node is responsible are checked and an attempt is made to restart any that are in a down state.

Whether a file system is a subject of a recovery attempt is determined by the max replication values for the file system. If the **mmfsfs -M** or **-R** value is greater than one, then the recovery code is executed. The recovery actions are asynchronous and GPFS will continue its processing while the recovery attempts take place. The results from the recovery actions and any errors that are encountered is recorded in the GPFS logs.

rpcPerfNumberDayIntervals

Controls the number of days that aggregated RPC data is saved. Every day the previous 24 hours of one-hour RPC data is aggregated into a one-day interval.

The default value for **rpcPerfNumberDayIntervals** is 30, which allows the previous 30 days of one-day intervals to be displayed. To conserve memory, fewer intervals can be configured to reduce the number of recent one-day intervals that can be displayed. The values that are allowed for **rpcPerfNumberDayIntervals** are in the range 4 - 60.

rpcPerfNumberHourIntervals

Controls the number of hours that aggregated RPC data is saved. Every hour the previous 60 minutes of one-minute RPC data is aggregated into a one-hour interval.

The default value for **rpcPerfNumberHourIntervals** is 24, which allows the previous day's worth of one-hour intervals to be displayed. To conserve memory, fewer intervals can be configured to reduce the number of recent one-hour intervals that can be displayed. The values that are allowed for **rpcPerfNumberHourIntervals** are 4, 6, 8, 12, or 24.

rpcPerfNumberMinuteIntervals

Controls the number of minutes that aggregated RPC data is saved. Every minute the previous 60 seconds of one-second RPC data is aggregated into a one-minute interval.

The default value for **rpcPerfNumberMinuteIntervals** is 60, which allows the previous hour's worth of one-minute intervals to be displayed. To conserve memory, fewer intervals can be configured to reduce the number of recent one-minute intervals that can be displayed. The values that are allowed for **rpcPerfNumberMinuteIntervals** are 4, 5, 6, 10, 12, 15, 20, 30, or 60.

mmchconfig

rpcPerfNumberSecondIntervals

Controls the number of seconds that aggregated RPC data is saved. Every second RPC data is aggregated into a one-second interval.

The default value for **rpcPerfNumberSecondIntervals** is 60, which allows the previous minute's worth of one-second intervals to be displayed. To conserve memory, fewer intervals can be configured to reduce the number of recent one-second intervals that can be displayed. The values that are allowed for **rpcPerfNumberSecondIntervals** are 4, 5, 6, 10, 12, 15, 20, 30, or 60.

rpcPerfRawExecBufferSize

Specifies the number of bytes to save in the buffer that stores raw RPC execution statistics. For each RPC received by a node, 16 bytes of associated data is saved in this buffer when the RPC completes. This circular buffer must be large enough to hold one second's worth of raw execution statistics.

The default value for **rpcPerfRawExecBufferSize** is 2M, which produces 131072 entries. Every second this data is processed, so the buffer should be 10% to 20% larger than what is needed to hold one second's worth of data.

rpcPerfRawStatBufferSize

Specifies the number of bytes to save in the buffer that stores raw RPC performance statistics. For each RPC sent to another node, 56 bytes of associated data is saved in this buffer when the reply is received. This circular buffer must be large enough to hold one second's worth of raw performance statistics.

The default value for **rpcPerfRawStatBufferSize** is 6M, which produces 112347 entries. Every second this data is processed, so the buffer should be 10% to 20% larger than what is needed to hold one second's worth of data.

sidAutoMapRangeLength

Controls the length of the reserved range for Windows SID to UNIX ID mapping. See "Identity management on Windows" in the *IBM Spectrum Scale: Advanced Administration Guide* for additional information.

sidAutoMapRangeStart

Specifies the start of the reserved range for Windows SID to UNIX ID mapping. See "Identity management on Windows" in the *IBM Spectrum Scale: Advanced Administration Guide* for additional information.

subnets

Specifies subnets used to communicate between nodes in a GPFS cluster or a remote GPFS cluster.

The subnets option must use the following format:

```
subnets="Subnet[/ClusterName[;ClusterName...][ Subnet[/ClusterName[;ClusterName...]]...]"
```

where:

ClusterName

Can be either a cluster name or a shell-style regular expression, which is used to match cluster names, such as:

CL[23].kgn.ibm.com

Matches **CL2.kgn.ibm.com** and **CL3.kgn.ibm.com**.

CL[0-7].kgn.ibm.com

Matches **CL0.kgn.ibm.com**, **CL1.kgn.ibm.com**, ... **CL7.kgn.ibm.com**.

CL*.ibm.com

Matches any cluster name that starts with **CL** and ends with **.ibm.com**.

CL?.kgn.ibm.com

Matches any cluster name that starts with **CL**, is followed by any one character, and then ends with **.kgn.ibm.com**.

The order in which you specify the subnets determines the order that GPFS uses these subnets to establish connections to the nodes within the cluster. GPFS observes the network setup of the operating system that includes the network mask for a specified subnet address. For example, **subnets="192.168.2.0"**, the mask may be anything from 23 bit (meaning that the subnet spans IP addresses 192.168.2.0 to 192.168.3.255) to 30 bit (meaning that the subnet spans IP addresses 192.168.2.0 to 192.168.2.3).

This feature cannot be used to establish fault tolerance or automatic failover. If the interface corresponding to an IP address in the list is down, GPFS does not use the next one on the list. For more information about subnets, see the *IBM Spectrum Scale: Advanced Administration Guide* and search on *Using remote access with public and private IP addresses*.

Specifying a cluster name or a cluster name pattern for each subnet is only needed when a private network is shared across clusters. If the use of a private network is confined within the local cluster, then no cluster name is required in the subnet specification.

systemLogLevel

Specifies the minimum severity level for messages sent to the system log. The severity levels from highest to lowest priority are: **alert**, **critical**, **error**, **warning**, **notice**, **configuration**, **informational**, **detail**, and **debug**. The value specified for this attribute can be any severity level, or the value **none** can be specified so no messages are sent to the system log. The default value is **error**.

GPFS generates some critical log messages that are always sent to the system logging service. This attribute only affects messages originating in the GPFS daemon (**mmfsd**). Log messages originating in some administrative commands will only be stored in the GPFS log file.

This attribute is only valid for Linux nodes.

tiebreakerDisks

Controls whether GPFS will use the node quorum with tiebreaker algorithm in place of the regular node-based quorum algorithm. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search for "node quorum with tiebreaker". To enable this feature, specify the names of one to three disks. Separate the NSD names with semicolon (;) and enclose the list in quotes. The disks do not have to belong to any particular file system, but must be directly accessible from the quorum nodes. For example:

```
tiebreakerDisks="gpfs1nsd;gpfs2nsd;gpfs3nsd"
```

To disable this feature, use:

```
tiebreakerDisks=no
```

Changing **tiebreakerDisks** is allowed while GPFS is running. However, if the traditional server-based (non-CCR) configuration repository is used, then when changing the **tiebreakerDisks**, GPFS must be down on all nodes in the cluster.

uidDomain

Specifies the UID domain name for the cluster.

GPFS must be down on all the nodes when changing the **uidDomain** attribute.

See the IBM white paper entitled *UID Mapping for GPFS in a Multi-cluster Environment* in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/SSFKCN/com.ibm.cluster.gpfs.doc/gpfs_uid/uid_gpfs.html).

unmountOnDiskFail

Controls how the daemon responds when it detects a disk failure:

yes The local node force-unmounts the file system that contains the failed disk. Other file

mmchconfig

systems on the local node and all nodes in the cluster continue to function normally, if they can. The local node can remount the file system when the disk problem is resolved. Use this setting in the following cases:

- You are using SAN-attached disks in large multinode configurations and you are not using replication.
- You have a node that hosts **descOnly** disks. See *Establishing disaster recovery for your GPFS cluster* in the *IBM Spectrum Scale: Advanced Administration Guide*.

no The daemon marks the disk as failed, notifies all nodes that use this disk that it has failed, and continues as long as it can without using the disk. You can make the disk active again with the **mmchdisk** command. This setting is appropriate when the node is using metadata-and-data replication, because the cluster can work from the replica until the failed disk is active again.

meta This option is like **No** except that the file system remains mounted unless it cannot access any replica of the metadata.

The **-N** flag is valid for this attribute.

usePersistentReserve

Specifies whether to enable or disable Persistent Reserve (PR) on the disks. Valid values are **yes** or **no** (**no** is the default). GPFS must be stopped on all nodes when setting this attribute.

To enable PR and to obtain recovery performance improvements, your cluster requires a specific environment:

- All disks must be PR-capable.
- On AIX, all disks must be hdisks; on Linux, they must be generic (**/dev/sd***) or DM-MP (**/dev/dm-***) disks.
- If the disks have defined NSD servers, all NSD server nodes must be running the same operating system (AIX or Linux).
- If the disks are SAN-attached to all nodes, all nodes in the cluster must be running the same operating system (AIX or Linux).

For more information, see “Reduced recovery time using Persistent Reserve” in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

verbsPorts

Specifies the InfiniBand device names and port numbers used for RDMA transfers between an NSD client and server. You must enable **verbsRdma** to enable **verbsPorts**.

The format for **verbsPorts** is:

```
verbsPorts="Device/Port/Fabric[ Device/Port/Fabric ...]"
```

In this format, *Device* is the HCA device name (such as *mtxca0* or *m1x4_0*); *Port* is the one-based port number (such as 1 or 2); and *Fabric* is a value to identify different InfiniBand (IB) fabrics (IB subnets on different switches).

If you do not specify a port number, GPFS uses port 1 as the default. If the fabric number is not specified, the fabric number is 0.

For example:

```
verbsPorts="m1x4_0/1/7 m1x4_0/2/8"
```

will create two RDMA connections between the NSD client and server using both ports of a dual ported adapter with a fabric identifier 7 on port 1 and fabric identifier 8 on port 2.

Another example, without the fabric number:

```
verbsPorts="mtxca0/1 mtxca0/2"
```

will create two RDMA connections between the NSD client and server using both ports of a dual ported adapter, with the fabric identifier defaulting to 0.

A third example, without port or fabric number:

```
verbsPorts="mlx4_0 mlx4_1"
```

will use port 1 on each HCA device for the RDMA connections.

The **-N** flag is valid for this attribute.

verbsRdma

Enables or disables InfiniBand RDMA using the Verbs API for data transfers between an NSD client and NSD server. Valid values are **enable** or **disable**.

The **-N** flag is valid for this attribute.

verbsRdmaCm

Enables or disables the RDMA Connection Manager (RDMA CM or RDMA_CM) using the RDMA_CM API for establishing connections between an NSD client and NSD server. Valid values are **enable** or **disable**. You must enable **verbsRdma** to enable **verbsRdmaCm**.

If RDMA CM is enabled for a node, the node will only be able to establish RDMA connections using RDMA CM to other nodes with **verbsRdmaCm** enabled. RDMA CM enablement requires IPoIB (IP over InfiniBand) with an active IP address for each port. Although IPv6 must be enabled, the GPFS implementation of RDMA CM does not currently support IPv6 addresses, so an IPv4 address must be used.

If **verbsRdmaCm** is not enabled when **verbsRdma** is enabled, the older method of RDMA connection will prevail.

The **-N** flag is valid for this attribute.

verbsRdmaRoCEToS

Specifies the Type of Service (ToS) value for clusters using RDMA over Converged Ethernet (RoCE). Acceptable values for this parameter are 0, 8, 16, and 24. The default value is -1.

If the user-specified value is neither the default nor an acceptable value, the script will exit with an error message to indicate that no change has been made. However, a RoCE cluster will continue to operate with an internally set ToS value of 0 even if the **mmchconfig** command failed. Different ToS values can be set for different nodes or groups of nodes.

The **-N** flag is valid for this attribute.

verbsRdmaSend

Enables or disables the use of InfiniBand RDMA rather than TCP for most GPFS daemon-to-daemon communication. When disabled, only data transfers between an NSD client and NSD server are eligible for RDMA. Valid values are **enable** or **disable**. The default value is **disable**. The **verbsRdma** option must be enabled for **verbsRdmaSend** to have any effect.

verbsRdmAsPerConnection

Sets the maximum number of simultaneous RDMA data transfer requests allowed per connection. The default value is zero; however, if this option is defaulted or set to zero, a value of 8 is used.

verbsRdmAsPerNode

Sets the maximum number of simultaneous RDMA data transfer requests allowed per node. The default value is zero; however, if this option is defaulted or set to zero, a value equal to the **nsdMaxWorkerThreads** setting is used.

verbsSendBufferMemoryMB

Sets the amount of page pool memory (in MiB) to reserve as dedicated buffer space for use by the **verbsRdmaSend** feature. If the value is unreasonably small or large (for example, larger than **pagepool**), the actual memory used is adjusted to a more appropriate value. If the value is zero

mmchconfig

(the default), a value is calculated based on the maximum number of RDMA's allowed per node (**verbsRdmasPerNode**). This option has no effect unless **verbsRdmaSend** is enabled.

workerThreads

Controls an integrated group of variables that tune file system performance. Use this variable to tune file systems in environments that are capable of high sequential or random read/write workloads or small-file activity. For new installations of the product, this variable is preferred over **worker1Threads** and **preFetchThreads**.

Important: If you set **workerThreads** to a non-default value, do not set **worker1Threads**.

The default value is 48. The valid range is 1-8192. However, the maximum value of **workerThreads** plus **preFetchThreads** plus **nsdMaxWorkerThreads** is 8192. The **-N** flag is valid with this variable.

This variable controls both internal and external variables. The internal variables include maximum settings for concurrent file operations, for concurrent threads that flush dirty data and metadata, and for concurrent threads that prefetch data and metadata. You can further adjust the external variables with the **mmchconfig** command:

logBufferCount

preFetchThreads

worker3Threads

The second variable is described in this help topic. The first and third variables are described in the article "Tuning Parameters" in the IBM Spectrum Scale wiki in developerWorks®. See <https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/General%20Parallel%20File%20System%20%28GPFS%29/page/Tuning%20Parameters>.

worker1Threads

Controls the maximum number of concurrent file operations at any one instant. If there are more requests than that, the excess will wait until a previous request has finished.

Important: If you set **workerThreads** to a non-default value, do not set **worker1Threads**.

This attribute is primarily used for random read or write requests that cannot be pre-fetched, random I/O requests, or small file activity. The default value is 48. The minimum value is 1. The maximum value of **prefetchThreads** plus **worker1Threads** plus **nsdMaxWorkerThreads** is 8192 on all 64-bit platforms.

The **-N** flag is valid for this attribute.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchconfig** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To change the maximum file system block size allowed to 4 MB, issue this command:

```
mmchconfig maxblocksize=4M
```

The system displays information similar to:

```
Verifying GPFS is stopped on all nodes ...
mmchconfig: Command successfully completed
mmchconfig: Propagating the cluster configuration data to all
  affected nodes. This is an asynchronous process.
```

To confirm the change, issue this command:

```
mmlsconfig
```

The system displays information similar to:

```
Configuration data for cluster ib.cluster:
```

```
-----
clusterName ib.cluster
clusterId 13882433899463047326
autoload no
minReleaseLevel 4.1.0.0
dmapiFileHandleSize 32
maxblocksize 4M
pagepool 2g
[c21f1n18]
pagepool 5g
[common]
verbsPorts mthca0/1
verbsRdma enable
subnets 10.168.80.0
adminMode central
```

```
File systems in cluster ib.cluster:
```

```
-----
/dev/fs1
```

See also

- “mmaddnode command” on page 245
- “mmchnode command” on page 381
- “mmcrcluster command” on page 403
- “mmdelnode command” on page 460
- “mmlsconfig command” on page 526
- “mmlscluster command” on page 524

Location

```
/usr/lpp/mmfs/bin
```

mmchdisk command

Changes state or parameters of one or more disks in a GPFS file system.

Synopsis

```
mmchdisk Device {resume | start} -a  
[-N {Node[,Node...]} | NodeFile | NodeClass}  
[--inode-criteria CriteriaFile]  
[-o InodeResultFile]
```

or

```
mmchdisk Device {suspend | empty | resume | stop | start | change}  
{-d "DiskDesc[;DiskDesc...]" | -F StanzaFile}  
[-N {Node[,Node...]} | NodeFile | NodeClass}  
[--inode-criteria CriteriaFile]  
[-o InodeResultFile]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchdisk** command to change the state or the parameters of one or more disks in a GPFS file system.

The state of a disk is a combination of its status and availability, displayed with the **mmlsdisk** command. Disk status is normally either **ready**, **emptied**, **suspended**, or **to be emptied**. A transitional status such as **replacing**, **replacement**, **to be emptied**, **suspended** or **being emptied** might also appear if a disk is being deleted or replaced. An emptied disk indicates that there are no blocks allocated on the disk and neither will any blocks get allocated from the disk. Running the **mmlsdisk** command will show the disk status as **emptied** and will be removed faster without the metadata scan. A suspended or being emptied disk is one that the user has decided not to place any new data on. Existing data on a suspended or being emptied disk may still be read or updated. Typically, a disk is suspended prior to restriping a file system. Suspending a disk tells the **mmrestripefs** command that data is to be migrated off that disk. Disk availability is either **up** or **down**.

Be sure to use **stop** before you take a disk offline for maintenance. You should also use **stop** when a disk has become temporarily inaccessible due to a disk failure that is repairable without loss of data on that disk (for example, an adapter failure or a failure of the disk electronics).

The *Disk Usage* (**dataAndMetadata**, **dataOnly**, **metadataOnly**, or **descOnly**) and *Failure Group* parameters of a disk are adjusted with the **change** option. See the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search for "recoverability considerations". The **mmchdisk change** command does not move data or metadata that resides on the disk. After changing disk parameters, in particular, *Disk Usage*, you may have to issue the **mmrestripefs** command with the **-r** option to relocate data so that it conforms to the new disk parameters.

The **mmchdisk** command can be issued for a mounted or unmounted file system. When maintenance is complete or the failure has been repaired, use the **mmchdisk** command with the **start** option. If the failure cannot be repaired without loss of data, you can use the **mmdeldisk** command.

Note:

1. The **mmchdisk** command cannot be used to change the NSD servers associated with the disk. Use the **mmchnsd** command for this purpose.

2. Similarly, the **mmchdisk** command cannot be used to change the storage pool for the disk. Use the **mmdeaddisk** and **mmadddisk** commands to move a disk from one storage pool to another.

Prior to GPFS 3.5, the disk information for the **mmchdisk** change option was specified in the form of disk descriptors defined as follows (with the second, third, sixth and seventh fields reserved):

```
DiskName:::DiskUsage:FailureGroup:::
```

For backward compatibility, the **mmchdisk** command will still accept the traditional disk descriptors, but their use is discouraged.

Parameters

Device

The device name of the file system to which the disks belong. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

suspend

or

empty

Instructs GPFS to stop allocating space on the specified disk. Place a disk in this state when you are preparing to restripe the file system off this disk because of faulty performance. This is a user-initiated state that GPFS never uses without an explicit command to change disk state. Existing data on a suspended disk may still be read or updated.

A disk remains in a suspended or to be emptied state until it is explicitly resumed. Restarting GPFS or rebooting nodes does not restore normal access to a suspended disk.

resume

Informs GPFS that a disk previously suspended is now available for allocating new space. If the disk is currently in a stopped state, it remains stopped until you specify the **start** option. Otherwise, normal read and write access to the disk resumes.

stop

Instructs GPFS to stop any attempts to access the specified disks. Use this option to tell the file system manager that a disk has failed or is currently inaccessible because of maintenance.

A disk remains stopped until it is explicitly started by the **mmchdisk** command with the **start** option.

You cannot run **mmchdisk stop** on a file system with replication 1.

start

Informs GPFS that disks previously stopped are now accessible. This is accomplished by first changing the disk availability from **down** to **recovering**. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was **down**) are repaired. If this operation is successful, the availability is then changed to **up**. If the metadata scan fails, availability is set to **unrecovered**. This could occur if too many other disks are **down**. The metadata scan can be re-initiated at a later time by issuing the **mmchdisk start** command again.

If more than one disk in the file system is down, they must all be started at the same time by issuing the **mmchdisk Device start -a** command. If you start them separately and metadata is stored on any disk that remains down, the **mmchdisk start** command fails.

change

Instructs GPFS to change the disk usage parameter, the failure group parameter, or both, according to the values specified in the NSD stanzas.

-d "DiskDesc[;DiskDesc...]"

A descriptor for each disk to be changed.

mmchdisk

Specify only disk names when using the **suspend**, **resume**, **stop**, or **start** options. Delimit multiple disk names with semicolons and enclose the list in quotation marks. For example, **"gpfs1nsd;gpfs2nsd"**

When using the **change** option, include the disk name and any new *Disk Usage* and *Failure Group* positional parameter values in the descriptor. Delimit descriptors with semicolons and enclose the list in quotation marks; for example, **"gpfs1nsd::**dataOnly**;gpfs2nsd::**metadataOnly:12"**.**

The use of disk descriptors is discouraged.

-F *StanzaFile*

Specifies a file containing the NSD stanzas for the disks to be changed. NSD stanzas have this format:

```
%nsd:  
  nsd=NsdName  
  usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}  
  failureGroup=FailureGroup  
  pool=StoragePool  
  servers=ServerList  
  device=DiskName
```

where:

nsd=*NsdName*

The name of the NSD to change. For a list of disks that belong to a particular file system, issue the **mmlsnsd -f Device**, **mmlsfs Device -d**, or **mmlsdisk Device** command. The **mmlsdisk Device** command will also show the current disk usage and failure group values for each of the disks. This clause is mandatory for the **mmchdisk** command.

usage={**dataOnly** | **metadataOnly** | **dataAndMetadata** | **descOnly**}

Specifies the type of data to be stored on the disk:

dataAndMetadata

Indicates that the disk contains both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disk contains data and does not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disk contains metadata and does not contain data.

descOnly

Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain disaster-recovery configurations. For more information, see the *IBM Spectrum Scale: Advanced Administration Guide* and search on "Synchronous mirroring utilizing GPFS replication"

This clause is meaningful only for the **mmchdisk change** option.

failureGroup=*FailureGroup*

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

pool=*StoragePool*

Specifies the storage pool to which the disk is to be assigned. If this name is not provided, the default is **system**.

Only the system storage pool can contain **metadataOnly**, **dataAndMetadata**, or **descOnly** disks. Disks in other storage pools must be **dataOnly**.

servers=*ServerList*

A comma-separated list of NSD server nodes. This clause is ignored by the **mmaddisk** command.

device=*DiskName*

The block device name of the underlying disk device. This clause is ignored by the **mmaddisk** command.

- a Specifies to change the state of all of the disks belonging to the file system, *Device*. This operand is valid only on the **resume** and **start** options.
- N {*Node* [,*Node* ...] | *NodeFile* | *NodeClass* }
Specifies a list of nodes that should be used for making the requested disk changes. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--inode-criteria *CriteriaFile*

Specifies the interesting inode criteria flag, where *CriteriaFile* is one of the following:

BROKEN

Indicates that a file has a data block with all of its replicas on disks that have been removed.

Note: **BROKEN** is always included in the list of flags even if it is not specified.

dataUpdateMiss

Indicates that at least one data block was not updated successfully on all replicas.

exposed

Indicates an inode with an exposed risk; that is, the file has data where all replicas are on suspended disks. This could cause data to be lost if the suspended disks have failed or been removed.

i11Compressed

Indicates an inode in which file compression or decompression is deferred, or in which a compressed file is partly decompressed to allow the file to be written into or memory-mapped.

i11Placed

Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

i11Replicated

Indicates that the file has a data block that does not meet the setting for the replica.

metaUpdateMiss

Indicates that there is at least one metadata block that has not been successfully updated to all replicas.

unbalanced

Indicates that the file has a data block that is not well balanced across all the disks in all failure groups.

mmchdisk

Note: If a file matches *any* of the specified interesting flags, all of its interesting flags (even those not specified) will be displayed.

-o *InodeResultFile*

Contains a list of the inodes that met the interesting inode flags that were specified on the **--inode-criteria** parameter. The output file contains the following:

INODE_NUMBER

This is the inode number.

DISKADDR

Specifies a dummy address for later **tsfindinode** use.

SNAPSHOT_ID

This is the snapshot ID.

ISGLOBAL_SNAPSHOT

Indicates whether or not the inode is in a global snapshot. Files in the live file system are considered to be in a global snapshot.

INDEPENDENT_FSETID

Indicates the independent fileset to which the inode belongs.

MEMO (INODE_FLAGS FILE_TYPE [ERROR])

Indicates the inode flag and file type that will be printed:

Inode flags:

BROKEN
exposed
dataUpdateMiss
i11Compressed
i11Placed
i11Replicated
metaUpdateMiss
unbalanced

File types:

BLK_DEV
CHAR_DEV
DIRECTORY
FIFO
LINK
LOGFILE
REGULAR_FILE
RESERVED
SOCK
UNLINKED
DELETED

Notes:

1. An error message will be printed in the output file if an error is encountered when repairing the inode.
2. **DISKADDR**, **ISGLOBAL_SNAPSHOT**, and **FSET_ID** work with the **tsfindinode** tool (`/usr/lpp/mmfs/bin/tsfindinode`) to find the file name for each inode. **tsfindinode** uses the output file to retrieve the file name for each interesting inode.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchdisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To **suspend** active disk **gpfs2nsd**, issue this command:

```
mmchdisk fs0 suspend -d gpfs2nsd
```

To confirm the change, issue this command:

```
mmlsdisk fs0
```

In IBM Spectrum Scale versions earlier than V4.1.1, the product displays information similar to the following example:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
gpfs2nsd	nsd	512	2	yes	yes	suspended	up	system
hd3vsdn01	nsd	512	2	yes	yes	ready	up	system
hd27n01	nsd	512	8	yes	yes	ready	up	system
hd28n01	nsd	512	8	yes	yes	ready	up	system
hd29n01	nsd	512	8	yes	yes	ready	up	system
hd10vsdn09	nsd	512	4003	no	yes	ready	up	sp1
hd11vsdn10	nsd	512	4003	no	yes	ready	up	sp1

Note: In product versions earlier than V4.1.1, the **mmlsdisk** command lists the disk status as suspended. In product versions V4.1.1 and later, the **mmlsdisk** command lists the disk status as to be emptied with both **mmchdisk suspend** or **mmchdisk empty** commands.

2. To **empty** active disk **gpfs1nsd**, issue this command:

```
mmchdisk fs0 empty -d gpfs1nsd
```

To confirm the change, issue this command:

```
mmlsdisk fs0 -L
```

In product version V4.1.1 and later, the system displays information similar to the following example:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	disk id	storage pool
gpfs1nsd	nsd	512	-1	Yes	Yes	to be emptied	up	1	system
gpfs2nsd	nsd	512	-1	Yes	Yes	to be emptied	up	2	system
gpfs3nsd	nsd	512	-1	Yes	Yes	ready	up	3	system
gpfs4nsd	nsd	512	-1	Yes	Yes	ready	up	4	system

Number of quorum disks: 3

Read quorum value: 2

Write quorum value: 2

Attention: Due to an earlier configuration change the file system may contain data that is at risk of being lost.

3. To specify that metadata should no longer be stored on disk **gpfs1nsd**, issue this command:

```
mmchdisk fs0 change -d "gpfs1nsd:::dataOnly"
```

To confirm the change, issue this command:

```
mmlsdisk fs0
```

mmchdisk

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
hd2vsdn01	nsd	512	2	yes	yes	ready	up	system
hd3vsdn01	nsd	512	2	yes	yes	ready	up	system
hd27n01	nsd	512	8	yes	yes	ready	up	system
gpfs1nsd	nsd	512	8	no	yes	ready	up	system
hd29n01	nsd	512	8	yes	yes	ready	up	system
hd10vsdn09	nsd	512	4003	no	yes	ready	up	sp1
hd11vsdn10	nsd	512	4003	no	yes	ready	up	sp1

4. To start a disk and check for files matching the interesting inode criteria located on the disk, issue this command:

```
mmchdisk fs1 start -d vmip2_nsd3 /tmp/crit --inode-criteria
```

The system displays information similar to:

```
mmnsddiscover: Attempting to rediscover the disks. This may take a while ...
mmnsddiscover: Finished.
vmip2.gpfs.net: GPFS: 6027-1805 [N] Rediscovered nsd server access to vmip2_nsd3.
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
Scanning file system metadata for data storage pool
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...
100.00 % complete on Wed Apr 15 10:20:37 2015 65792 inodes with total 398 MB data processed)
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-3312 No inode was found matching the criteria.
```

The disk was started successfully. No files matching the requested criteria were found.

See also

- *Displaying GPFS disk states in the IBM Spectrum Scale: Advanced Administration Guide.*
- “mmaddisk command” on page 239
- “mmchnsd command” on page 388
- “mmdeldisk command” on page 449
- “mmlsdisk command” on page 528
- “mmlsnsd command” on page 549
- “mmrpldisk command” on page 648

Location

```
/usr/lpp/mmfs/bin
```

mmcheckquota command

Checks file system user, group and fileset quotas.

Synopsis

```
mmcheckquota [-v] [-N {Node[,Node...] | NodeFile | NodeClass}]
              [--qos QosClass] {-a | Device [Device ...]}
```

or

```
mmcheckquota {-u UserQuotaFile | -g GroupQuotaFile | -j FilesetQuotaFile}
              [--qos QosClass] Device
```

or

```
mmcheckquota --backup backupDir Device
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

The **mmcheckquota** command serves two purposes:

1. Count inode and space usage in a file system by user, group and fileset, and write the collected data into quota files.

Note: In cases where small files do not have an additional block allocated for them, quota usage may show less space usage than expected.

2. Replace either the user, group, or fileset quota files, for the file system designated by *Device*, thereby restoring the quota files for the file system. These files must be contained in the root directory of *Device*. If a backup copy does not exist, an empty file is created when the **mmcheckquota** command is issued.

The **mmcheckquota** command counts inode and space usage for a file system and writes the collected data into quota files. Indications leading you to the conclusion you should run the **mmcheckquota** command include:

- **MMFS_QUOTA** error log entries. This error log entry is created when the quota manager has a problem reading or writing the quota file.
- Quota information is lost due to a node failure. A node failure could leave users unable to open files or deny them disk space that their quotas should allow.
- The in-doubt value is approaching the quota limit.

The sum of the in-doubt value and the current usage may not exceed the hard limit. Consequently, the actual block space and number of files available to the user of the group may be constrained by the in-doubt value. If the in-doubt value approaches a significant percentage of the quota, use the **mmcheckquota** command to account for the lost space and files.

- User, group, or fileset quota files are corrupted.

The **mmcheckquota** command is I/O-intensive and should be run when the system load is light. When issuing the **mmcheckquota** command on a mounted file system, negative in-doubt values may be reported if the quota server processes a combination of up-to-date and back-level information. This is a transient situation and can be ignored.

If a file system is ill-replicated, the **mmcheckquota** command will not be able to determine exactly how many valid replicas actually exist for some of the blocks. If this happens, the used block count results from **mmcheckquota** will not be accurate. It is recommended that you run **mmcheckquota** to restore

mmcheckquota

accurate usage count after the file system is no longer ill-replicated.

Parameters

-a Checks all GPFS file systems in the cluster from which the command is issued.

--backup *BackupDirectory*

Specifies a backup directory, which must be in the same GPFS file system as the root directory of *Device*.

In IBM Spectrum Scale V4.1.1 and later, you can use this parameter to copy quota files. The command copies three quota files to the specified directory.

Device

Specifies the device name of the file system. File system names do not need to be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

-g *GroupQuotaFileName*

Replaces the current group quota file with the file indicated.

When replacing quota files with the **-g** option, the quota file must be in the root directory of the GPFS file system.

-j *FilesetQuotaFilename*

Replaces the current fileset quota file with the file indicated.

When replacing quota files with the **-j** option, the quota file must be in the root directory of the GPFS file system.

-N *{Node[,Node...] | NodeFile | NodeClass}*

Specifies the nodes that will participate in a parallel quota check of the system. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-u *UserQuotaFilename*

Replaces the current user quota file with the file indicated.

When replacing quota files with the **-u** option, the quota file must be in the root directory of the GPFS file system.

--qos *QoSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Options

-v Reports discrepancies between calculated and recorded disk quotas.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcheckquota** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmcheckquota** command is issued.

Examples

1. To check quotas for file system **fs0**, issue this command:

```
mmcheckquota fs0
```

The system displays information only if a problem is found.

2. To check quotas for all file systems, issue this command:

```
mmcheckquota -a
```

The system displays information only if a problem is found or if quota management is not enabled for a file system:

```
fs2: no quota management installed
```

```
fs3: no quota management installed
```

3. To report discrepancies between calculated and recorded disk quotas, issue this command:

```
mmcheckquota -v fs1
```

The system displays information similar to:

```
fs1: Start quota check
1 % complete on Fri Apr 17 13:07:47 2009
6 % complete on Fri Apr 17 13:07:48 2009
11 % complete on Fri Apr 17 13:07:49 2009
17 % complete on Fri Apr 17 13:07:50 2009
22 % complete on Fri Apr 17 13:07:51 2009
28 % complete on Fri Apr 17 13:07:52 2009
33 % complete on Fri Apr 17 13:07:53 2009
38 % complete on Fri Apr 17 13:07:54 2009
44 % complete on Fri Apr 17 13:07:55 2009
49 % complete on Fri Apr 17 13:07:56 2009
55 % complete on Fri Apr 17 13:07:57 2009
61 % complete on Fri Apr 17 13:07:58 2009
66 % complete on Fri Apr 17 13:07:59 2009
72 % complete on Fri Apr 17 13:08:00 2009
78 % complete on Fri Apr 17 13:08:01 2009
83 % complete on Fri Apr 17 13:08:02 2009
89 % complete on Fri Apr 17 13:08:03 2009
94 % complete on Fri Apr 17 13:08:04 2009
Finished scanning the inodes for fs1.
Merging results from scan.
fs1: quota check found the following differences:
USR 0: 288400 subblocks counted (was 288466); 24 inodes counted (was 81)
USR 60011: 50 subblocks counted (was 33); 2 inodes counted (was 20)
USR 60012: 225 subblocks counted (was 223); 9 inodes counted (was 4)
USR 60013: 175 subblocks counted (was 146); 7 inodes counted (was 26)
```

mmcheckquota

```
USR 60014: 200 subblocks counted (was 178); 8 inodes counted (was 22)
USR 60015: 275 subblocks counted (was 269); 11 inodes counted (was 0)
USR 60019: 0 subblocks counted (was 9); 0 inodes counted (was 5)
USR 60020: 0 subblocks counted (was 1); 0 inodes counted (was 3)
GRP 0: 28845098 subblocks counted (was 28844639); 14 inodes counted (was 91)
FILESET 0: 28849125 subblocks counted (was 28848717); 105 inodes counted (was 24)
```

See also

- “mmedquota command” on page 481
- “mmfsck command” on page 487
- “mmlsquota command” on page 559
- “mmquotaon command” on page 619
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

/usr/lpp/mmfs/bin

mmchfileset command

Changes the attributes of a GPFS fileset.

Synopsis

```
mmchfileset Device {FilesetName | -J JunctionPath}
                [-j NewFilesetName] [-t NewComment] [-p afmAttribute=Value...]
                [--allow-permission-change PermissionChangeMode]
                [--inode-limit MaxNumInodes[:NumInodesToPreallocate]]
                [--iam-mode Mode]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmchfileset** command changes attributes for an existing GPFS fileset.

For information on GPFS filesets, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system that contains the fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

FilesetName

Specifies the name of the fileset.

-J *JunctionPath*

Specifies the junction path name for the fileset.

A junction is a special directory entry that connects a name in a directory of one fileset to the root directory of another fileset.

-j *NewFilesetName*

Specifies the new name that is to be given to the fileset. This name must be less than 256 characters in length. The root fileset cannot be renamed.

-t *NewComment*

Specifies an optional comment that appears in the output of the **mmlsfileset** command. This comment must be less than 256 characters in length. This option cannot be used on the root fileset.

-p *afmAttribute=Value*

Specifies an AFM configuration attribute and its value. More than one **-p** option can be specified.

The following AFM configuration attributes are valid:

afmAsyncDelay

Specifies (in seconds) the amount of time by which write operations are delayed (because write operations are asynchronous with respect to remote clusters). For write-intensive applications that keep writing to the same set of files, this delay is helpful because it replaces multiple writes to the home cluster with a single write containing the latest data. However, setting a very high value weakens the consistency of data on the remote cluster.

This configuration parameter is applicable only for writer caches (SW, IW, and Primary) in which data from cache is pushed to home.

Valid values are 1 through 2147483647. The default is 15.

mmchfileset

afmDirLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a directory, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of that directory has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 60. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmDirOpenRefreshInterval

Controls the frequency of data revalidations that are triggered by such I/O operations as **read** or **write** (specified in seconds). After a directory has been cached, **open** requests resulting from I/O operations on that object are directed to the cached directory until the specified amount of time has passed. Once the specified amount of time has passed, the **open** request gets directed to a gateway node rather than to the cached directory.

Valid values are 0 through 2147483647. The default is 60. Setting a lower value guarantees a higher level of consistency.

afmEnableAutoEviction

Enables eviction on a given fileset. A **yes** value specifies that eviction is allowed on the fileset. A **no** value specifies that eviction is not allowed on the fileset.

See also the topic about cache eviction in the *IBM Spectrum Scale: Advanced Administration Guide*.

afmExpirationTimeout

Is used with **afmDisconnectTimeout** (which can be set only through **mmchconfig**) to control how long a network outage between the cache and home clusters can continue before the data in the cache is considered out of sync with home. After **afmDisconnectTimeout** expires, cached data remains available until **afmExpirationTimeout** expires, at which point the cached data is considered expired and cannot be read until a reconnect occurs.

Valid values are 0 through 2147483647. The default is 300.

afmFileLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a file, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of the file has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 30. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmFileOpenRefreshInterval

Controls the frequency of data revalidations that are triggered by such I/O operations as **read** or **write** (specified in seconds). After a file has been cached, **open** requests resulting from I/O operations on that object are directed to the cached file until the specified amount of time has passed. Once the specified amount of time has passed, the **open** request gets directed to a gateway node rather than to the cached file.

Valid values are 0 through 2147483647. The default is 30. Setting a lower value guarantees a higher level of consistency.

afmMode

Specifies the mode in which the cache operates. Valid values are the following:

single-writer | sw

Specifies single-writer mode.

read-only | ro

Specifies read-only mode. (For **mmcrfileset**, this is the default value.)

local-updates | lu

Specifies local-updates mode.

independent-writer | iw

Specifies independent-writer mode.

Primary

Specifies the primary mode for AFM asynchronous data replication.

Secondary

Specifies the secondary mode for AFM asynchronous data replication.

Changing from single-writer/read-only modes to read-only/local-updates/single-writer is supported. When changing from read-only to single-writer, the read-only cache is up-to-date. When changing from single-writer to read-only, all requests from cache should have been played at home. Changing from local-updates to read-only/local-updates/single-writer is restricted. A typical dataset is set up to include a single cache cluster in single-writer mode (which generates the data) and one or more cache clusters in local-updates or read-only mode. AFM single-writer/independent-writer filesets can be converted to primary using the conversion process described in the chapter about AFM disaster recovery in the *IBM Spectrum Scale: Advanced Administration Guide*. Primary/secondary filesets cannot be converted to AFM filesets.

In case of AFM asynchronous data replication, the **mmchfileset** command cannot be used to convert to primary from secondary. For information about converting to primary and secondary, see the chapter about AFM disaster recovery in the *IBM Spectrum Scale: Advanced Administration Guide*.

For more information, see the topic about caching modes in the *IBM Spectrum Scale: Advanced Administration Guide* chapter about active file management.

afmNumFlushThreads

Defines the number of threads used on each gateway to synchronize updates to the home cluster. The default value is 4, which is sufficient for most installations. The current maximum value is 1024, which is too high for most installations; setting this parameter to such an extreme value should be avoided.

afmNumReadThreads

Defines the number of threads that can be used on each participating gateway node during parallel read. The default value of this parameter is 1; that is, one reader thread will be active on every gateway node for each big write operation qualifying for splitting per the parallel read threshold value. The valid range of values is 1 to 64.

afmNumWriteThreads

Defines the number of threads that can be used on each participating gateway node during parallel write. The default value of this parameter is 1; that is, one writer thread will be active on every gateway node for each big write operation qualifying for splitting per the parallel write threshold value. Valid values can range from 1 to 64.

afmParallelReadChunkSize

Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel reads. Values are interpreted in terms of bytes. The default value of this parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelReadThreshold

Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default value is 1024 MB. The valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

mmchfileset

afmParallelWriteChunkSize

Defines the minimum chunk size of the write that needs to be distributed among the gateway nodes during parallel writes. Values are interpreted in terms of bytes. The default value of this parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelWriteThreshold

Defines the threshold beyond which parallel writes become effective. Writes are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default value of this parameter is 1024 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmPrefetchThreshold

Controls partial file caching and prefetching. Valid values are the following:

0 Enables full file prefetching. This is useful for sequentially accessed files that are read in their entirety, such as image files, home directories, and development environments. The file will be prefetched after three blocks have been read into the cache.

1-99

Specifies the percentage of file size that must be cached before the entire file is prefetched. A large value is suitable for a file accessed either randomly or sequentially but partially, for which it might be useful to ingest the rest of the file when most of it has been accessed.

100

Disables full file prefetching. This value only fetches and caches data that is read by the application. This is useful for large random-access files, such as databases, that are either too big to fit in the cache or are never expected to be read in their entirety. When all data blocks are accessed in the cache, the file is marked as cached.

0 is the default value.

For local-updates mode, the whole file is prefetched when the first update is made.

afmPrimaryId

Specifies the unique primary ID of the primary fileset for asynchronous data replication. This is used for connecting a secondary to a primary.

afmReadSparseThreshold

Specifies the size in MB for files in cache beyond which sparseness is maintained. For all files below the specified threshold, sparseness is not maintained.

afmRPO

Specifies the recovery point objective (RPO) interval in minutes for a primary fileset.

afmShowHomeSnapshot

Controls the visibility of the home snapshot directory in cache. For this to be visible in cache, this variable has to be set to **yes**, and the snapshot directory name in cache and home should not be the same.

yes

Specifies that the home snapshot link directory is visible.

no

Specifies that the home snapshot link directory is not visible.

See also the topic about peer snapshots in the *IBM Spectrum Scale: Advanced Administration Guide*.

afmTarget

The only allowed value is **disable**. It is used to convert AFM filesets to regular independent filesets; for example:

```
mmchfileset fs1 ro -p afmTarget=disable
```

Once an AFM fileset is converted to a regular fileset, it cannot be changed back to an AFM fileset again. Disabling of AFM is permitted only on RO and LU filesets.

--allow-permission-change *PermissionChangeMode*

Specifies the new permission change mode. This mode controls how **chmod** and ACL operations are handled on objects in the fileset. Valid modes are as follows:

chmodOnly

Specifies that only the UNIX change mode operation (**chmod**) is allowed to change access permissions (ACL commands and API will not be accepted).

setAc1Only

Specifies that permissions can be changed using ACL commands and API only (**chmod** will not be accepted).

chmodAndSetAc1

Specifies that **chmod** and ACL operations are permitted. If the **chmod** command (or **setattr** file operation) is issued, the result depends on the type of ACL that was previously controlling access to the object:

- If the object had a Posix ACL, it will be modified accordingly.
- If the object had an NFSv4 ACL, it will be replaced by the given UNIX mode bits.

Note: This is the default setting when a fileset is created.

chmodAndUpdateAc1

Specifies that **chmod** and ACL operations are permitted. If **chmod** is issued, the ACL will be updated by privileges derived from UNIX mode bits.

--inode-limit *MaxNumInodes[:NumInodesToPreallocate]*

Specifies the new inode limit for the inode space owned by the specified fileset. The *FilesetName* or *JunctionPath* must refer to an independent fileset. The *NumInodesToPreallocate* specifies an optional number of additional inodes to pre-allocate for the inode space. Use the **mmchfs** command to change inode limits for the root fileset.

The *MaxNumInodes* and *NumInodesToPreallocate* values can be specified with a suffix, for example 100K or 2M.

--iam-mode *Mode*

Specifies the integrated archive manager (IAM) mode for the fileset. IAM modes can be used to modify some of the file-operation restrictions that normally apply to immutable files. The following values (listed in order of strictness) are accepted:

- ad** | **advisory**
- nc** | **noncompliant**
- co** | **compliant**

For more information about IAM modes, see the topic about immutability and appendOnly restrictions in the Information Lifecycle Management chapter of *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmchfileset

Security

You must have root authority or be a fileset owner to run the **mmchfileset** command with the **-t** option. All other options require root authority.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

This command renames fileset **fset1** to **fset2** and gives it the comment "first fileset":

```
mmchfileset gpfs1 fset1 -j fset2 -t 'first fileset'
```

The system displays a message similar to:

```
Fileset 'fset1' changed.
```

To confirm the change, issue this command:

```
mmlsfileset gpfs1 -L
```

The system displays information similar to:

```
Filesets in file system 'gpfs1':  
Name Id RootInode ParentId Created InodeSpace MaxInodes AllocInodes Comment  
root 0 3 -- Mon Apr 12 16:31:05 2010 0 8001536 8001536 root fileset  
fset2 2 13569 0 Mon Apr 12 16:32:28 2010 0 0 0 first fileset
```

See also

- “mmchfs command” on page 371
- “mmcrfileset command” on page 408
- “mmdelfileset command” on page 455
- “mmlinkfileset command” on page 517
- “mmlsfileset command” on page 532
- “mmunlinkfileset command” on page 687

Location

```
/usr/lpp/mmfs/bin
```


mmchfs command

Changes the attributes of a GPFS file system.

Synopsis

```
mmchfs Device [-A {yes | no | automount}] [-D {posix | nfs4}] [-E {yes | no}]
[-k {posix | nfs4 | all}] [-K {no | whenpossible | always}]
[-L LogFileSize] [-m DefaultMetadataReplicas] [-n NumNodes]
[-o MountOptions] [-r DefaultDataReplicas] [-S {yes | no | relatime}]
[-T Mountpoint] [-t DriveLetter] [-V {full | compat}] [-z {yes | no}]
[--filesetdf | --nofilesetdf]
[--inode-limit MaxNumInodes[:NumInodesToPreallocate]]
[--log-replicas LogReplicas] [--mount-priority Priority]
[--perfileset-quota | --noperfileset-quota]
[--rapid-repair | --norapid-repair]
[--write-cache-threshold HAWCThreshold]
```

or

```
mmchfs Device -Q {yes | no}
```

or

```
mmchfs Device -W NewDeviceName
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchfs** command to change the attributes of a GPFS file system.

Parameters

Device

The device name of the file system to be changed.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**. However, file system names must be unique across GPFS clusters.

This must be the first parameter.

-A {yes | no | automount}

Indicates when the file system is to be mounted:

yes

When the GPFS daemon starts.

no Manual mount.

automount

On non-Windows nodes, when the file system is first accessed. On Windows nodes, when the GPFS daemon starts.

Note: The file system must be unmounted prior to changing the automount settings.

-D {nfs4 | posix}

Specifies whether a deny-write open lock will block writes, which is expected and required by NFS V4, Samba, and Windows. File systems supporting NFS V4 must have **-D nfs4** set. The option **-D posix** allows NFS writes even in the presence of a deny-write open lock. If you intend to export the file system using NFS V4 or Samba, or mount your file system on Windows, you must use **-D nfs4**. For NFS V3 (or if the file system is not NFS exported at all) use **-D posix**.

mmchfs

-E {yes | no}

Specifies whether to report exact **mtime** values. If **-E no** is specified, the **mtime** value is periodically updated. If you desire to always display exact modification times, specify **-E yes**.

-k {posix | nfs4 | all}

Specifies the type of authorization supported by the file system:

posix

Traditional GPFS ACLs only (NFS V4 and Windows ACLs are not allowed). Authorization controls are unchanged from earlier releases.

nfs4

Support for NFS V4 and Windows ACLs only. Users are not allowed to assign traditional GPFS ACLs to any file system objects (directories and individual files).

all

Any supported ACL type is permitted. This includes traditional GPFS (**posix**) and NFS V4 and Windows ACLs (**nfs4**).

The administrator is allowing a mixture of ACL types. For example, **fileA** may have a **posix** ACL, while **fileB** in the same file system may have an NFS V4 ACL, implying different access characteristics for each file depending on the ACL type that is currently assigned.

Avoid specifying **nfs4** or **all** unless files will be exported to NFS V4 or Samba clients, or the file system will be mounted on Windows. NFS V4 and Windows ACLs affect file attributes (mode) and have access and authorization characteristics that are different from traditional GPFS ACLs.

-K {no | whenpossible | always}

Specifies whether strict replication is to be enforced:

no Strict replication is not enforced. GPFS will try to create the needed number of replicas, but will still return EOK as long as it can allocate at least one replica.

whenpossible

Strict replication is enforced provided the disk configuration allows it. If there is only one failure group, strict replication will not be enforced.

always

Strict replication is enforced.

For more information, see the topic *Strict replication* in the *IBM Spectrum Scale: Problem Determination Guide*.

-L *LogFileSize*

Specifies the new size of the internal log files. The *LogFileSize* specified must be a multiple of the metadata block size. The minimum size is 256 KB and the maximum size is 1024 MB. Specify this value with the K or M character, for example: 8M.

You must restart the GPFS daemons before the new log file size takes effect. The GPFS daemons can be restarted one node at a time. When the GPFS daemon is restarted on the last node in the cluster, the new log size becomes effective.

-m *DefaultMetaDataReplicas*

Changes the default number of metadata replicas. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of *MaxMetaDataReplicas* set when the file system was created.

Changing the default replication settings using the **mmchfs** command does not change the replication setting of existing files. After running the **mmchfs** command, the **mmrestripefs** command with the **-R** option can be used to change *all* existing files or you can use the **mmchattr** command to change a small number of existing files.

-n NumNodes

Changes the number of nodes for a file system. This setting is *just an estimate* and will only be used to affect the layout of system metadata for storage pools created after the setting is changed.

-o MountOptions

Specifies the mount options to pass to the mount command when mounting the file system. For a detailed description of the available mount options, see the following *IBM Spectrum Scale: Administration and Programming Reference* topic: “GPFS-specific mount options” on page 28.

-Q {yes | no}

If **-Q yes** is specified, quotas are activated automatically when the file system is mounted. If **-Q no** is specified, the quota files remain in the file system, but are not used.

For more information, see the topic *Enabling and disabling GPFS quota management* in the *IBM Spectrum Scale: Advanced Administration Guide*.

-r DefaultDataReplicas

Changes the default number of data replicas. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of *MaxDataReplicas* set when the file system was created.

Changing the default replication settings using the **mmchfs** command does not change the replication setting of existing files. After running the **mmchfs** command, the **mmrestripefs** command with the **-R** option can be used to change *all* existing files or you can use the **mmchattr** command to change a small number of existing files.

-S {yes | no | relatime}

Suppress the periodic updating of the value of **atime** as reported by the **gpfs_stat()**, **gpfs_fstat()**, **stat()**, and **fstat()** calls. If **yes** is specified, these calls report the last time the file was accessed when the file system was mounted with **-S no**.

If **relatime** is specified, the file access time is updated only if the existing access time is older than the value of the **atimeDeferredSeconds** configuration attribute or the existing file modification time is greater than the existing access time.

-T Mountpoint

Change the mount point of the file system starting at the next mount of the file system.

The file system must be unmounted on all nodes prior to issuing the command.

-t DriveLetter

Changes the Windows drive letter for the file system.

The file system must be unmounted on all nodes prior to issuing the command.

-V {full | compat}

Changes the file system format to the latest format supported by the currently installed level of GPFS. This *may* cause the file system to become permanently incompatible with earlier releases of GPFS.

Note: The **-V** option cannot be used to make file systems created prior to GPFS 3.2.1.5 available to Windows nodes. Windows nodes can mount only file systems that are created with GPFS 3.2.1.5 or later.

Before issuing **-V**, see “Migration, coexistence and compatibility” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*. Ensure that all nodes in the cluster have been migrated to the latest level of GPFS code and that you have successfully run the **mmchconfig release=LATEST** command.

For information about specific file system format and function changes when you upgrade to the current release, see the topic *File system format changes between versions of GPFS* in the *IBM Spectrum Scale: Advanced Administration Guide*.

mmchfs

full

Enables all new functionality that requires different on-disk data structures. Nodes in remote clusters running an older GPFS version will no longer be able to mount the file system. If there are any nodes running an older GPFS version that have the file system mounted at the time the command is issued, the **mmchfs** command will fail.

compat

Enables only backward-compatible format changes. Nodes in remote clusters that are running GPFS 3.5 will still be able to mount the file system. Nodes running GPFS 3.4 or earlier will no longer be able to mount the file system.

-W *NewDeviceName*

Assign *NewDeviceName* to be the device name for the file system.

-z {yes | no}

Enable or disable DMAPI on the file system. Turning this option on will require an external data management application such as Tivoli Storage Manager (TSM) hierarchical storage management (HSM) before the file system can be mounted. For further information on DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

--filesetdf

Specifies that when quotas are enforced for a fileset, the numbers reported by the **df** command are based on the quotas for the fileset (rather than the entire file system). This option affects the **df** command behavior only on Linux nodes.

--nofilesetdf

Specifies that when quotas are enforced for a fileset, the numbers reported by the **df** command are based on the quotas for the entire file system (rather than individual filesets).

--inode-limit *MaxNumInodes[:NumInodesToPreallocate]*

MaxNumInodes specifies the maximum number of files that can be created. Allowable values range from the current number of created inodes (determined by issuing the **mmdf** command with **-F**), through the maximum number of files possibly supported as constrained by the formula:

maximum number of files = (total file system space) / (inode size + subblock size)

Note: This formula works only for simpler configurations. For complex configurations, such as separation of data and metadata, this formula might not provide an accurate result.

If your file system has additional disks added or the number of inodes was insufficiently sized at file system creation, you can change the number of inodes and hence the maximum number of files that can be created.

For file systems that will be doing parallel file creates, if the total number of free inodes is not greater than 5% of the total number of inodes, there is the potential for slowdown in file system access. Take this into consideration when changing your file system.

NumInodesToPreallocate specifies the number of inodes that will be pre-allocated by the system right away. If this number is not specified, GPFS allocates inodes dynamically as needed.

The *MaxNumInodes* and *NumInodesToPreallocate* values can be specified with a suffix, for example 100K or 2M. Note that in order to optimize file system operations, the number of inodes that are actually created may be greater than the specified value.

This option applies only to the root fileset. When there are multiple inode spaces, use the **--inode-space** option of the **mmchfileset** command to alter the inode limits of independent filesets. The **mmchfileset** command can also be used to modify the root inode space. The **--inode-space** option of the **mmlsfs** command shows the sum of all inode spaces.

--log-replicas *LogReplicas*

Specifies the number of recovery log replicas. Valid values are **1**, **2**, **3**, or **DEFAULT**. If **DEFAULT** is

specified, the number of log replicas is the same as the number of metadata replicas currently in effect for the file system and will change when this number is changed.

Changing the default replication settings using the **mmchfs** command does not change the replication setting of existing files. After running the **mmchfs** command, the **mmrestripefs** command with the **-R** option can be used to change existing log files.

This option is only applicable if the recovery log is stored in the **system.log** storage pool.

--mount-priority *Priority*

Controls the order in which the individual file systems are mounted at daemon startup or when one of the **all** keywords is specified on the **mmmount** command.

File systems with higher *Priority* numbers are mounted after file systems with lower numbers. File systems that do not have mount priorities are mounted last. A value of zero indicates no priority.

--perfileset-quota

Sets the scope of user and group quota limit checks to the individual fileset level (rather than the entire file system). Before you activate or deactivate per-fileset quotas, you must unmount the file system from the cluster.

--noperfileset-quota

Sets the scope of user and group quota limit checks to the entire file system (rather than per individual filesets).

--rapid-repair

Keeps track of incomplete replication on an individual file block basis (as opposed to the entire file). This may result in a faster repair time when very large files are only partially ill-replicated.

--norapid-repair

Specifies that replication status is kept on a whole file basis (rather than on individual block basis).

--write-cache-threshold *HAWCThreshold*

Specifies the maximum length (in bytes) of write requests that will be initially buffered in the highly-available write cache before being written back to primary storage. Only synchronous write requests are guaranteed to be buffered in this fashion.

A value of 0 disables this feature. 64K is the maximum supported value. Specify in multiples of 4K.

This feature can be enabled or disabled at any time (the file system does not need to be unmounted). For more information about this feature, see the topic *Highly-available write cache (HAWC)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmchfs

Examples

To change the default replicas for metadata to 2 and the default replicas for data to 2 for new files created in the **fs0** file system, issue this command:

```
mmchfs fs0 -m 2 -r 2
```

To confirm the change, issue this command:

```
mmisfs fs0 -m -r
```

The system displays information similar to:

flag value	description
-m 2	Default number of metadata replicas
-r 2	Default number of data replicas

See also

- “mmchfileset command” on page 365
- “mmcrfs command” on page 414
- “mmdelfs command” on page 458
- “mmdf command” on page 470
- “mmfsck command” on page 487
- “mmlsfs command” on page 536
- “mmrestripefs command” on page 642

Location

```
/usr/lpp/mmfs/bin
```

mmchlicense command

Controls the type of GPFS license associated with the nodes in the cluster.

Synopsis

```
mmchlicense {client|fpo|server} [--accept] -N {Node[,Node...] | NodeFile | NodeClass}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchlicense** command to change the type of GPFS license associated with the nodes in the cluster.

For information on GPFS license designation, see “GPFS license designation” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Parameters

client | fpo | server

The type of GPFS license to be assigned to the nodes specified with the **-N** parameter.

client

The IBM Spectrum Scale Client license permits exchange of data between nodes that locally mount the same GPFS file system. No other export of the data is permitted. The GPFS client may not be used for nodes to share GPFS data directly through any application, service, protocol or method, such as Network File System (NFS), Common Internet File System (CIFS), File Transfer Protocol (FTP), or Hypertext Transfer Protocol (HTTP). For these functions, an IBM Spectrum Scale Server license would be required.

fpo

The IBM Spectrum Scale FPO license permits the licensed node to perform NSD server functions for sharing GPFS data with other nodes that have an IBM Spectrum Scale FPO or IBM Spectrum Scale Server license. This license cannot be used to share data with nodes that have an IBM Spectrum Scale Client license or non-GPFS nodes.

server

The IBM Spectrum Scale Server license permits the licensed node to perform GPFS management functions such as cluster configuration manager, quorum node, manager node, and Network Shared Disk (NSD) server. In addition, the IBM Spectrum Scale Server license permits the licensed node to share GPFS data directly through any application, service protocol or method such as NFS, CIFS, FTP, or HTTP.

--accept

Indicates that you accept the applicable licensing terms. The license acceptance prompt will be suppressed.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes that are to be assigned the specified license type.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

mmchlicense

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchlicense** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To designate nodes k145n04 and k145n05 as possessing a GPFS server license, issue this command:

```
mmchlicense server --accept -N k145n04,k145n05
```

The system displays information similar to:

The following nodes will be designated as possessing GPFS server licenses:

```
k145n04.kgn.ibm.com
```

```
k145n05.kgn.ibm.com
```

```
mmchlicense: Command successfully completed
```

```
mmchlicense: Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

See also

- “mmlslicense command” on page 540

Location

```
/usr/lpp/mmfs/bin
```


mmchmgr command

Assigns a new file system manager node or cluster manager node.

Synopsis

```
mmchmgr {Device | -c} [Node]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmchmgr** command assigns a new file system manager node or cluster manager node.

Parameters

Device

The device name of the file system for which the file system manager node is to be changed. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

-c Changes the cluster manager node.

Node

The target node to be appointed as either the new cluster manager node or the new file system manager node. Target nodes for manager functions are selected according to the following criteria:

- Target nodes for the cluster manager function must be specified from the list of quorum nodes.
- Target nodes for the file system manager function should be specified from the list of manager nodes, although this is not strictly required.

If *Node* is not specified, the new manager is selected automatically.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchmgr** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. Assume the file system manager for the file system **gpfs1** is currently **k164n05**. To migrate the file system manager responsibilities to **k164n06**, issue this command:

```
mmchmgr gpfs1 k164n06
```

The system displays information similar to:

mmchmgr

```
GPFS: 6027-628 Sending migrate request to current manager node 89.116.68.69 (k164n05).
GPFS: 6027-629 [N] Node 89.116.68.69 (k164n05) resigned as manager for gpfs1.
GPFS: 6027-630 [N] Node 89.116.68.70 (k164n06) appointed as manager for gpfs1.
```

To verify the change, issue the command:

```
mmlsmgr gpfs1
```

The system displays information similar to:

```
file system manager node [from 89.116.68.69 (k164n06)]
-----
gpfs1 89.116.68.69 (k164n06)
```

2. To change the cluster manager node, issue the command:

```
mmchmgr -c c5n107
```

The system displays information similar to:

```
Appointing node 9.114.132.107 (c5n107) as cluster manager
Node 9.114.132.107 (c5n107) has taken over as cluster manager
```

To verify the change, issue the command:

```
mmlsmgr -c
```

The system displays information similar to:

```
Cluster manager node: 9.114.132.107 (c5n107)
```

See also

- “mmlsmgr command” on page 542

Location

```
/usr/lpp/mmfs/bin
```

mmchnode command

Changes node attributes.

Synopsis

```
mmchnode change-options -N {Node[,Node...]} | NodeFile | NodeClass}
```

or

```
mmchnode {-S Filename | --spec-file=Filename}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchnode** command to change one or more attributes on a single node or on a set of nodes. If conflicting node designation attributes are specified for a given node, the last value is used. If any of the attributes represent a node-unique value, the **-N** option must resolve to a single node.

Do not use the **mmchnode** command to change the gateway node role while IO is happening on the fileset. Run the **flushpending** command to flush any pending messages from queues before running the **mmchnode** command for the gateway node role changes.

Parameters

-N {Node[,Node...]} | NodeFile | NodeClass}

Specifies the nodes whose states are to be changed.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-S *Filename* | --spec-file=*Filename*

Specifies a file with a detailed description of the changes to be made. Each line represents the changes to an individual node and has the following format:

```
node-identifier change-options
```

change-options

A blank-separated list of attribute[=*value*] pairs. The following attributes can be specified:

--admin-interface={hostname | ip_address}

Specifies the name of the node to be used by GPFS administration commands when communicating between nodes. The admin node name must be specified as an IP address or a hostname that is resolved by the host command to the desired IP address. If the keyword **DEFAULT** is specified, the admin interface for the node is set to be equal to the daemon interface for the node.

--client

Specifies that the node should not be part of the pool of nodes from which cluster managers, file system managers, and token managers are selected.

--ces-enable

Enables Cluster Export Services (CES) on the node.

--ces-disable

Disables CES on the node.

--ces-group=Group[,Group...]

Adds one or more groups to the specified nodes.

mmchnode

--noces-group=Group[,Group...]

Removes one or more groups from the specified nodes.

--cnfs-disable

Disables the CNFS functionality of a CNFS member node.

--cnfs-enable

Enables a previously-disabled CNFS member node.

--cnfs-groupid=groupid

Specifies a failover recovery group for the node. If the keyword DEFAULT is specified, the CNFS recovery group for the node is set to zero.

For additional information, refer to "Implementing a clustered NFS using GPFS on Linux" in the *IBM Spectrum Scale: Advanced Administration Guide*.

--cnfs-interface=ip_address_list

A comma-separated list of host names or IP addresses to be used for GPFS cluster NFS serving.

The specified IP addresses can be real or virtual (aliased). These addresses must be configured to be static (not DHCP) and to not start at boot time.

The GPFS daemon interface for the node cannot be a part of the list of CNFS IP addresses.

If the keyword DEFAULT is specified, the CNFS IP address list is removed and the node is no longer considered a member of CNFS.

If **adminMode** central is in effect for the cluster, all CNFS member nodes must be able to execute remote commands without the need for a password.

For additional information, refer to "Implementing a clustered NFS using GPFS on Linux" in the *IBM Spectrum Scale: Advanced Administration Guide*.

--daemon-interface={hostname | ip_address}

Specifies the host name or IP address to be used by the GPFS daemons for node-to-node communication. The host name or IP address must refer to the communication adapter over which the GPFS daemons communicate. Alias interfaces are not allowed. Use the original address or a name that is resolved by the host command to the original address.

Before you specify this option, you must stop GPFS on all the nodes in the cluster. You cannot use the keyword DEFAULT with this option.

You cannot specify the **--daemon-interface** option for a quorum node if CCR is enabled.

Temporarily change the node to a nonquorum node. Then run the **mmchnode** command with the **--daemon-interface** option against the nonquorum node. Finally, change the node back into a quorum node.

--gateway | --nogateway

Specifies whether the node is to be designated as a gateway node or not.

--manager

Designates the node as part of the pool of nodes from which file system managers and token managers are selected.

--nonquorum

Designates the node as a non-quorum node. If two or more quorum nodes are downgraded at the same time, GPFS must be stopped on all nodes in the cluster. GPFS does not have to be stopped if the nodes are downgraded one at a time.

--perfmon | --noperfmon

Specifies whether the node is to be designated as a perfmon node or not.

--nosnmp-agent

Stops the SNMP subagent and specifies that the node should no longer serve as an SNMP collector node. For additional information see the topic "GPFS SNMP support" in the *IBM Spectrum Scale: Advanced Administration Guide*.

--quorum

Designates the node as a quorum node.

Note: If you are designating a node as a quorum node, and **adminMode central** is in effect for the cluster, you must ensure that GPFS is up and running on that node (**mmgetstate** reports the state of the node as **active**).

--snmp-agent

Designates the node as an SNMP collector node. If the GPFS daemon is active on this node, the SNMP subagent will be started as well. For additional information see the topic "GPFS SNMP support" in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchnode** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

- To change nodes k145n04 and k145n05 to be both quorum and manager nodes, issue this command:

```
mmchnode --quorum --manager -N k145n04,k145n05
```

The system displays information similar to:

```
Wed May 16 04:50:24 EDT 2007: mmchnode: Processing node k145n04.kgn.ibm.com
Wed May 16 04:50:24 EDT 2007: mmchnode: Processing node k145n05.kgn.ibm.com
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

After completion, **mmlscluster** displays information similar to:

```
GPFS cluster information
=====
GPFS cluster name: mynodes.kgn.ibm.com
GPFS cluster id: 680681553700098206
GPFS UID domain: mynodes.kgn.ibm.com
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
-----
Primary server: k145n04.kgn.ibm.com
Secondary server: k145n06.kgn.ibm.com
```

```
Node Daemon node name IP address Admin node name Designation
-----
```

mmchnode

```
1 k145n04.kgn.ibm.com 9.114.68.68 k145n04.kgn.ibm.com quorum-manager
2 k145n05.kgn.ibm.com 9.114.68.69 k145n05.kgn.ibm.com quorum-manager
3 k145n06.kgn.ibm.com 9.114.68.70 k145n06.kgn.ibm.com
```

2. To change nodes k145n04 and k145n05 to be both quorum and manager nodes, and node k45n06 to be a non-quorum node, issue this command:

```
mmchnode -S /tmp/specFile
```

Where the contents of /tmp/specFile are:

```
k145n04 --quorum --manager
k145n05 --quorum --manager
k145n06 --nonquorum
```

The system displays information similar to:

```
Wed May 16 05:23:31 EDT 2007: mmchnode: Processing node k145n04
Wed May 16 05:23:32 EDT 2007: mmchnode: Processing node k145n05
Wed May 16 05:23:32 EDT 2007: mmchnode: Processing node k145n06
Verifying GPFS is stopped on all nodes ...
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

And **mmlscluster** displays information similar to:

GPFS cluster information

```
=====
GPFS cluster name: mynodes.kgn.ibm.com
GPFS cluster id: 680681553700098206
GPFS UID domain: mynodes.kgn.ibm.com
Remote shell command: /usr/bin/rsh
Remote file copy command: /usr/bin/rcp
```

GPFS cluster configuration servers:

```
-----
Primary server: k145n04.kgn.ibm.com
Secondary server: k145n06.kgn.ibm.com
```

Node	Daemon node name	IP address	Admin node name	Designation
1	k145n04.kgn.ibm.com	9.114.68.68	k145n04.kgn.ibm.com	quorum-manager
2	k145n05.kgn.ibm.com	9.114.68.69	k145n05.kgn.ibm.com	quorum-manager
3	k145n06.kgn.ibm.com	9.114.68.70	k145n06.kgn.ibm.com	

See also

- “mmchconfig command” on page 331
- “mmlscluster command” on page 524

Location

/usr/lpp/mmfs/bin

mmchnodeclass command

Changes user-defined node classes.

Synopsis

```
mmchnodeclass ClassName {add | delete | replace}
               -N {Node[,Node...] | NodeFile | NodeClass}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmchnodeclass** command to make changes to existing user-defined node classes.

Parameters

ClassName

Specifies the name of an existing user-defined node class to modify.

add

Adds the nodes specified with the **-N** option to *ClassName*.

delete

Deletes the nodes specified with the **-N** option from *ClassName*.

replace

Replaces all *ClassName* members with a new list of nodes specified with the **-N** option.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the member names of nodes and node classes that will be used for the **add**, **delete**, or **replace** action.

NodeClass cannot be used to add members that already contain other node classes. For example, two user-defined node classes called **siteA** and **siteB** were used to create a new node class called **siteAandB**, as follows:

```
mmcrnodeclass siteAandB -N siteA,siteB
```

The **siteAandB** node class cannot later be specified for *NodeClass* when adding to existing node classes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchnodeclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmchnodeclass

Examples

To display the current members of a user-defined node class called **siteA**, issue this command:

```
mmfnsnodeclass siteA
```

The system displays information similar to:

Node Class Name	Members
siteA	c8f2c4vp1,c8f2c4vp2

To add node **c8f2c1vp4** to the member list of the user-defined node class **siteA**, issue this command:

```
mmchnodeclass siteA add -N c8f2c1vp4
```

The system displays information similar to:

```
mmchnodeclass: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

To display the updated member list of **siteA**, issue this command:

```
mmfnsnodeclass siteA
```

The system displays information similar to:

Node Class Name	Members
siteA	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2

To delete node **c8f2c4vp2** from the member list of **siteA**, issue this command:

```
mmchnodeclass siteA delete -N c8f2c4vp2
```

The system displays information similar to:

```
mmchnodeclass: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

To display the updated member list of **siteA**, issue this command:

```
mmfnsnodeclass siteA
```

The system displays information similar to:

Node Class Name	Members
siteA	c8f2c1vp4,c8f2c4vp1

To replace all the current members of **siteA** with the members of node class **linuxNodes**, issue this command:

```
mmchnodeclass siteA replace -N linuxNodes
```

The system displays information similar to:

```
mmchnodeclass: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

To display the updated member list of **siteA**, issue this command:

```
mmfnsnodeclass siteA
```

The system displays information similar to:

Node Class Name	Members
siteA	linuxNodes

See also

- “mmcrnodeclass command” on page 424
- “mmdelnodeclass command” on page 463
- “mmlsnodeclass command” on page 546

Location

/usr/lpp/mmfs/bin

mmchnsd command

Changes Network Shared Disk (NSD) configuration attributes.

Synopsis

```
mmchnsd {"DiskDesc[;DiskDesc...]" | -F StanzaFile}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmchnsd** command serves several purposes. You can use it to:

- Specify a server list for an NSD that does not have one.
- Change the NSD server nodes specified in the server list.
- Delete the server list. The disk must now be SAN-attached to all nodes in the cluster on which the file system will be mounted.

You must follow these rules when changing NSDs:

- Identify the disks by the NSD names that were given to them by the **mmcrnsd** command.
- Explicitly specify values for all NSD servers on the list even if you are only changing one of the values.
- Unmount the file system that contains the NSD being changed prior to issuing the **mmchnsd** command.
- Connect the NSD to the new nodes prior to issuing the **mmchnsd** command.
- **mmchnsd** cannot change the disk usage or failure group for an NSD. Use the **mmchdisk** command to change these attributes.
- To move a disk from one storage pool to another, use the **mmdeldisk** and **mmadddisk** commands.
- You cannot change the name of the NSD.

Prior to GPFS 3.5, the disk information was specified in the form of disk descriptors defined as:

```
DiskName:ServerList:
```

For backward compatibility, the **mmchnsd** command will still accept the traditional disk descriptors but their use is discouraged.

Parameters

DiskDesc

A descriptor for each NSD to be changed. Each descriptor is separated by a semicolon (;). The entire list must be enclosed in single or double quotation marks. The use of disk descriptors is discouraged.

-F *StanzaFile*

Specifies a file containing the NSD stanzas for the disks to be changed. NSD stanzas have this format:

```
%nsd:  
  nsd=NsdName  
  servers=ServerList  
  usage=DiskUsage  
  failureGroup=FailureGroup  
  pool=StoragePool  
  device=DiskName
```

where:

nsd=NsName

Is the NSD name that was given to the disk by the **mmcrnsd** command. This clause is mandatory for the **mmchnsd** command.

servers=ServerList

Is a comma-separated list of NSD server nodes. You can specify up to eight NSD servers in this list. The defined NSD will preferentially use the first server on the list. If the first server is not available, the NSD will use the next available server on the list.

When specifying server nodes for your NSDs, the output of the **mmlscluster** command lists the host name and IP address combinations recognized by GPFS. The utilization of aliased host names not listed in the **mmlscluster** command output may produce undesired results.

If you do not define a *ServerList*, GPFS assumes that the disk is SAN-attached to all nodes in the cluster. If all nodes in the cluster do not have access to the disk, or if the file system to which the disk belongs is to be accessed by other GPFS clusters, you must specify a value for *ServerList*.

To remove the NSD server list, do not specify a value for *ServerList* (remove or comment out the **servers=ServerList** clause of the NSD stanza).

usage=DiskUsage

Specifies the type of data to be stored on the disk. If this clause is specified, the value must match the type of usage already in effect for the disk; **mmchnsd** cannot be used to change this value.

failureGroup=FailureGroup

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

If this clause is specified, the value must match the failure group already in effect for the disk; **mmchnsd** cannot be used to change this value.

pool=StoragePool

Specifies the storage pool to which the disk is to be assigned. If this clause is specified, the value must match the storage pool already in effect for the disk; **mmchnsd** cannot be used to change this value.

device=DiskName

Is the block device name of the underlying disk device. This clause is ignored by the **mmchnsd** command.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmchnsd

Security

You must have root authority to run the **mmchnsd** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

If the disk **gpfs1nsd** is currently defined with **k145n05** as the first server and **k145n07** as the second server, and you want to replace **k145n05** with **k145n09**, create a file `./newNSDstanza` that contains:

```
%nsd: nsd=gpfs1nsd
      servers=k145n09,k148n07
```

Issue this command:

```
mmchnsd -F ./newNSDstanza
```

To confirm the changes, issue this command:

```
mmlnsd -d gpfs1nsd
```

The system displays information similar to:

```
File system  Disk name  NSD servers
-----
fs2          gpfs1nsd  k145n09.ppd.pok.ibm.com,k145n07.ppd.pok.ibm.com
```

See also

- “mmchdisk command” on page 354
- “mmcrcluster command” on page 403
- “mmcrnsd command” on page 426
- “mmlnsd command” on page 549

Location

```
/usr/lpp/mmfs/bin
```

mmchpolicy command

Establishes policy rules for a GPFS file system.

Synopsis

```
mmchpolicy Device PolicyFilename [-t DescriptiveName] [-I {yes | test}]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmchpolicy** command to establish the rules for policy-based lifecycle management of the files in a given GPFS file system. Some of the things that can be controlled with the help of policy rules are:

- File placement at creation time
- Replication factors
- Movement of data between storage pools
- File deletion

The **mmapplypolicy** command must be run to move data between storage pools or delete files.

Policy changes take effect immediately on all nodes that have the affected file system mounted. For nodes that do not have the file system mounted, policy changes take effect upon the next mount of the file system.

For file systems that are created at or upgraded to product version V4.1.1 or later: If there are no **SET POOL** policy rules installed to a file system by **mmchpolicy**, the system acts as if the single rule **SET POOL 'first-data-pool'** is in effect, where *first-data-pool* is the firstmost non-system pool that is available for file data storage, if such a non-system pool is available. (“Firstmost” is the first according to an internal index of all pools.) However, if there are no policy rules installed and there is no non-system pool, the system acts as if **SET POOL 'system'** is in effect.

This change applies only to file systems that were created at or upgraded to V4.1.1 or later. Until a file system is upgraded, if no **SET POOL** rules are present (set by **mmchpolicy**) for the file system, all data will be stored in the **'system'** pool.

For information on GPFS policies, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

Specifies the device name of the file system for which policy information is to be established or changed. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**. This must be the first parameter.

PolicyFilename

Specifies the name of the file that contains the policy rules. If you specify **DEFAULT**, GPFS replaces the current policy file with a single policy rule that assigns all newly-created files to the **system** storage pool.

Options

-I {yes | test}

Specifies whether to activate the rules in the policy file *PolicyFileName*.

mmchpolicy

yes

The policy rules are validated and immediately activated. This is the default.

test

The policy rules are validated, but not installed.

-t *DescriptiveName*

Specifies an optional descriptive name to be associated with the policy rules. The string must be less than 256 characters in length. If not specified, the descriptive name defaults to the base name portion of the *PolicyFileName* parameter.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchpolicy** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This command validates a policy before it is installed:

```
mmchpolicy fs2 fs2.pol -I test
```

The system displays output similar to:

```
Validated policy `fs2.pol': parsed 3 Placement Rules, 0 Restore Rules,  
3 Migrate/Delete/Exclude Rules, 0 List Rules, 0 External Pool/List Rules
```

2. This command installs a policy:

```
mmchpolicy fs2 fs2.pol
```

The system displays output similar to:

```
Validated policy `fs2.pol': parsed 1 Placement Rules, 0 Restore Rules,  
0 Migrate/Delete/Exclude Rules, 1 List Rules, 1 External Pool/List Rules  
Policy `fs2.pol' installed and broadcast to all nodes.
```

To confirm the change, issue this command:

```
mmlspolicy fs2
```

The system displays output similar to:

```
Policy file for file system '/dev/fs2':  
  Installed by root@k155n11.kgn.ibm.com on Mon Dec 12  
  16:56:31 2005.  
  First line from original file 'fs2.pol' was:  
/* This is the policy for the fs2 GPFS file system. */
```

See also

- “mmapplypolicy command” on page 266
- “mmlspolicy command” on page 552

Location

/usr/lpp/mmfs/bin

mmchpool

mmchpool command

Modifies storage pool properties.

Synopsis

```
mmchpool Device {PoolName[,PoolName...] | all}
           [--block-group-factor BlockGroupFactor]
           [--write-affinity-depth WriteAffinityDepth]
```

or

```
mmchpool Device -F PoolDescriptorFile
```

Availability

Available on all IBM Spectrum Scale editions.

When running the **mmchpool** command, the file system must be unmounted on all nodes.

Description

Use the **mmchpool** command to change storage pool properties.

Parameters

Device

Specifies the device name of the file system for which storage pool information is to be changed. File system names do not need to be fully qualified; for example, `fs0` is as acceptable as `/dev/fs0`.

PoolName[,PoolName...]

Specifies one or more storage pools for which attributes will be changed.

all

Changes the attributes for all the storage pools in the specified file system.

--block-group-factor *BlockGroupFactor*

Specifies how many file system blocks are laid out sequentially on disk to behave like a single large block. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

--write-affinity-depth *WriteAffinityDepth*

Specifies the allocation policy to be used. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

-F *PoolDescriptorFile*

Specifies a file used to describe the storage pool attributes. The file contains one line per storage pool, in the following format:

```
%pool:name:blockSize:diskUsage:reserved:maxDiskSize:allocationType:allowWriteAffinity:writeAffinityDepth:blockGroupFactor:
```

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmchpool** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Example

For example, to change the **writeAffinityDepth** to 2 for FPO pool pool1 of file system fs1, issue this command:

```
mmchpool fs1 pool1 --write-affinity-depth 2
```

To confirm the change, issue this command:

```
mmlspool fs1 pool1 -L
```

The system displays information similar to the following:

```
# mmlspool fs_1 p1 -L
Pool:
  name           = pool1
  poolID         = 65537
  blockSize      = 4 MB
  usage          = dataOnly
  maxDiskSize    = 11 TB
  layoutMap      = cluster
  allowWriteAffinity = yes
  writeAffinityDepth = 2
  blockGroupFactor = 128
```

See also

- “mmlspool command” on page 554

Location

```
/usr/lpp/mmfs/bin
```

mmchqos command

Changes the Quality of Service for I/O operations (QoS) settings for a file system.

Synopsis

```
mmchqos Device --enable [--reset] [--force]
           [pool=PoolName[,QoSClass={nnnIOPS | unlimited}][,QoSClass={nnnIOPS | unlimited}] ...]
```

or

```
mmchqos Device --disable
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Run the **mmchqos** command to allocate shares of IOPS to the two QoS classes:

- A **maintenance** class for I/O-intensive, potentially long-running GPFS commands. Typically you assign fewer IOPS to this class to prevent the I/O-intensive commands from dominating file system performance and significantly delaying other tasks.
- An **other** class for all other processes. Typically you assign more IOPS or **unlimited** to this class so that normal processes have greater access to I/O resources and finish more quickly.

When QoS is enabled, it restricts the active processes in a QoS class from collectively consuming more than the number of IOPS that you allocate to the class. It queues further I/O attempts until more I/O operations become available.

Remember the following points:

- You can allocate shares of IOPS separately for each storage pool.
- QoS divides each IOPS allocation evenly among the active nodes in the cluster.
- Allocations persist across unmounting and remounting the file system.
- QoS stops applying allocations when you unmount the file system and resumes when you remount it.
- When you change allocations or mount the file system, a brief delay due to reconfiguration occurs before QoS starts applying allocations.

For more information about this command, see *Setting the Quality of Service for I/O operations (QoS)* in *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

Device

The device name of the file system to which the command applies.

--enable

Causes QoS to start or to continue applying IOPS allocations. If you are specifying this option for the first time, then QoS sets any QoS classes that you do not specify in the command to **unlimited** IOPS. On subsequent enables, by default, QoS sets only the IOPS allocations that you specify in the command. It does not disturb other IOPS allocations. Compare the **--reset** parameter.

--disable

Causes QoS to stop applying IOPS allocations. Lets the file system run without any participation by QoS.

--reset

Causes QoS to set any QoS classes that you do not specify in the command to **unlimited** IOPS.

--force

Causes QoS to accept an IOPS value lower than 100 IOPS.

PoolName

Specifies a storage pool to whose QoS classes the IOPS are allocated. If you specify an asterisk (*) as the pool name, then the IOPS are allocated to the QoS classes of the *residual pools*. The residual pools are storage pools that you have not specified by name in any **mmchqos** command.

QoSClass=nnnIOPS

Identifies a QoS class and allocates IOPS to it. QoS divides this allocation evenly among the active nodes in the cluster. You can specify the following classes:

maintenance

The class of I/O-intensive, potentially slow-running GPFS administration commands. See the following list. Typically, you allocate fewer IOPS to this QoS class so that the commands that belong to it do not reduce overall file system performance.

These commands run in the **maintenance** class by default. But when you start a command, you can explicitly assign it to either QoS class. In certain situations, you might assign a command to the **other** class so that it runs faster and completes sooner. The assignment is effective only for the instance of the command that you are starting.

mmadddisk
mmapplypolicy
mmbackup
mmcheckquota
mmdefragfs
mmdeldisk
mmdelfileset
mmdelsnapshot
mmdf
mmfileid
mmfsck
mmlssnapshot
mmrestripefs
mmrpldisk

other The class of all other processes that use I/O. Typically you assign more IOPS or **unlimited** to this class so that normal processes have greater access to I/O resources and finish more quickly.

You can use the following values for IOPS:

- A value in the range **0IOPS-1999999999IOPS**. For an IOPS value less than 100, you must specify the **-force** option. Otherwise, QoS displays an error message like the following one:
 maintenance=99iops is not reasonable. To insist, try --force.
- **unlimited**: QoS does not restrict access to I/O operations.

Exit status

0 Successful completion.

Nonzero

A failure occurred.

Security

You must have root authority to run the **mmchqos** command.

mmchqos

The node on which you enter the command must be able to execute remote shell commands on any other administration node in the cluster. It must be able to do so without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a GPFS file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. The following command enables QoS and allocates default IOPS values to the QoS classes **maintenance** and **other** in all storage pools. The default values depend on whether you previously enabled QoS. If so, then the QoS classes are set to their previous values. If not, then the classes are set to **unlimited**:

```
mmchqos fs0 --enable
```

2. The following command disables QoS but does not change the allocations of IOPS:

```
mmchqos fs0 --disable
```

3. The following command enables QoS and allocates 123 IOPS to the **maintenance** class of each residual pool. By default it sets the **other** class of each residual pool to **unlimited**:

```
mmchqos fs0 --enable pool=*,maintenance=123IOPS
```

4. The following command enables QoS and allocates 222 IOPS to the **maintenance** class of each residual pool. It also allocates 576 IOPS to the **maintenance** class of the pool mySSDs. By default it sets the **other** classes of each residual pool and mySSDs to **unlimited**. You might make an allocation like this one to favor a pool of high-speed storage (mySSDs) that you expect to be accessed frequently:

Storage pool	maintenance class	other class
Each residual pool	222IOPS	unlimited
mySSDs	567IOPS	unlimited

```
mmchqos fs0 --enable pool=*,maintenance=222IOPS pool=mySSDs,maintenance=567IOPS
```

5. The following command enables QoS and allocates IOPS to the classes of the residual pools and three named pools:

Storage pool	maintenance class	other class
Each residual pool	444IOPS	555IOPS
system	111IOPS	unlimited
second	222IOPS	unlimited
third	333IOPS	unlimited

The command is all on one line:

```
mmchqos fs0 --enable pool=*,maintenance=444IOPS,other=555iops  
pool=system,maintenance=111IOPS,other=unlimited  
pool=second,maintenance=222IOPS,other=unlimited  
pool=third,other=unlimited,maintenance=333IOPS
```

6. The following command enables QoS and allocates IOPS to the classes of three named pools. By default, it also sets both classes of each residual pool to **unlimited**:

Storage pool	maintenance class	other class
Each residual pool	unlimited	unlimited
system	111IOPS	unlimited
second	222IOPS	unlimited
third	333IOPS	unlimited

The command is all on one line:

```
mmchqos fs0 --enable pool=system,maintenance=111IOPS,other=unlimited
           pool=second,maintenance=222IOPS,other=unlimited
           pool=third,other=unlimited,maintenance=333IOPS
```

7. The following command enables QoS and allocates IOPS to both classes of the system pool. Also, because the command contains the **--reset** parameter, it sets both classes of all the other storage pools in the file system to **unlimited**. The reset affects not only any residual pools, but also any named pools that are not explicitly mentioned in this command.

Storage pool	maintenance class	other class
system	111IOPS	unlimited
Each residual pool, if any	unlimited	unlimited
Each named pool, if any, other than system	unlimited	unlimited

The command is all on one line:

```
mmchqos fs0 --enable --reset pool=system,maintenance=111IOPS,other=unlimited
```

8. The first part of the following command assigns IOPS to the QoS classes of the residual pools. It assigns 222 IOPS to the **maintenance** class of each residual pool. And by default, it assigns **unlimited** to the **other** class of each residual pool.

The second part of the command allocates 456 IOPS to the **other** class of the storage pool mySAN, rather than assigning it the default value **unlimited**. You might make an allocation like this one to a SAN controller that serves both GPFS and other systems.

Storage pool	maintenance class	other class
Each residual pool	222IOPS	unlimited
mySAN	123IOPS	456IOPS

The command is all on one line:

```
mmchqos fs0 --enable pool=*,maintenance=222IOPS
           pool=mySAN,other=456IOPS,maintenance=123IOPS
```

See also

- “mmlsqos command” on page 556
- “Setting the Quality of Service for I/O operations (QoS)” on page 40

Location

/usr/lpp/mmfs/bin

mmclone command

Creates and manages file clones.

Synopsis

mmclone snap *SourceFile* [*CloneParentFile*]

or

mmclone copy *CloneParentFile TargetFile*

or

mmclone split *Filename* [*Filename...*]

or

mmclone redirect *Filename* [*Filename...*]

or

mmclone show *Filename* [*Filename...*]

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmclone** command to create and manage file clones. Clones are writable snapshots of individual files. Cloning a file is similar to creating a copy of a file, but the creation process is faster and more space efficient because no additional disk space is consumed until the clone or the original file is modified. The keyword specified after **mmclone** determines which action is performed:

snap

Creates a read-only snapshot of an existing file for the purpose of cloning. This read-only snapshot becomes known as the clone parent.

If only one file is specified with the **mmclone snap** command, it will convert that file to a clone parent without creating a separate clone parent file. When using this method to create a clone parent, the specified file cannot be open for writing or have hard links.

copy

Creates a file clone from a clone parent created with the **mmclone snap** command or from a file in a snapshot.

split

Splits a file clone from all clone parents.

redirect

Splits a file clone from the immediate clone parent only.

show

Displays the current status for one or more specified files. When a file is a clone, the report will show the parent inode number. When a file was cloned from a file in a snapshot, **mmclone show** displays the snapshot and fileset information.

The Depth field in the **mmclone show** output denotes the distance of the file from the root of the clone tree of which it is a member. The root of a clone tree has depth 0. This field is blank if the file in question is not a clone. This field is not updated when a clone's ancestor is redirected or split from the clone tree. However, even if a clone's ancestor has been split or redirected, the depth of the clone should always be greater than that of each of its ancestors.

The maximum depth for a clone tree is 1000.

Note: The **mmclone** command does not copy extended attributes.

If a snapshot has file clones, those file clones should be deleted or split from their clone parents prior to deleting the snapshot. Use the **mmclone split** or **mmclone redirect** command to split file clones. Use a regular delete (**rm**) command to delete a file clone. If a snapshot is deleted that contains a clone parent, any attempts to read a block that refers to the missing snapshot will return an error. A policy file can be created to help determine if a snapshot has file clones.

For more information about file clones and policy files, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

SourceFile

Specifies the name of a file to clone.

CloneParentFile

When *CloneParentFile* is specified with a **mmclone snap** command, it indicates the name of the read-only clone parent that will be created from *SourceFile*.

When *CloneParentFile* is specified with a **mmclone copy** command, it indicates the name of a read-only clone parent. The *CloneParentFile* can be a clone parent created with the **mmclone snap** command or a file in a snapshot.

TargetFile

Specifies the name of the writable file clone that will be created from *CloneParentFile*.

Filename

Specifies the name of one or more files to **split**, **redirect**, or **show**.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

To run the **mmclone** command, you must have read access to the source file that will be cloned, and write access to the directory where the file clone will be created.

Examples

1. To create a clone parent called **base.img** from a file called **test01.img**, issue this command:

```
mmclone snap test01.img base.img
```

To use this clone parent to create a file clone called **test02.img**, issue this command:

```
mmclone copy base.img test02.img
```

After the file clone is created, use the **mmclone show** command to show information about all **img** files in the current directory:

```
mmclone show *.img
```

The system displays output similar to the following:

mmclone

Parent	Depth	Parent inode	File name
yes	0		base.img
no	1	148488	test01.img
no	1	148488	test02.img

2. To create a file clone called **file1.clone** from a file called **file1** in the **snap1** snapshot, issue this command:

```
mmclone copy /fs1/.snapshots/snap1/file1 file1.clone
```

See also

- “mmcrsnapshot command” on page 431
- “mmdelsnapshot command” on page 467

Location

```
/usr/lpp/mmfs/bin
```


mmcrcluster command

Creates a GPFS cluster from a set of nodes.

Synopsis

```
mmcrcluster -N {NodeDesc[,NodeDesc...] | NodeFile}
             [--ccr-enable | {--ccr-disable -p PrimaryServer [-s SecondaryServer]}]
             [ [-r RemoteShellCommand] [-R RemoteFileCopyCommand] | --use-sudo-wrapper ]
             [-C ClusterName] [-U DomainName] [-A]
             [-c ConfigFile | --profile ProfileName]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmcrcluster** command to create a GPFS cluster.

Upon successful completion of the **mmcrcluster** command, the `/var/mmfs/gen/mmsdrfs` and the `/var/mmfs/gen/mmfsNodeData` files are created on each of the nodes in the cluster. Do not delete these files under any circumstances. For more information, see “Quorum” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Follow these rules when creating your GPFS cluster:

- While a node may mount file systems from multiple clusters, the node itself may only be added to a single cluster using the **mmcrcluster** or **mmaddnode** command.
- The nodes must be available for the command to be successful. If any of the nodes listed are not available when the command is issued, a message listing those nodes is displayed. You must correct the problem on each node and issue the **mmaddnode** command to add those nodes.
- Designate at least one but not more than seven nodes as quorum nodes. How many quorum nodes altogether you will have depends on whether you intend to use the node quorum with tiebreaker algorithm or the regular node based quorum algorithm. For more information, see “Quorum” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- After the nodes are added to the cluster, use the **mmchlicense** command to designate appropriate GPFS licenses to the new nodes.
- Clusters that will include both UNIX and Windows nodes must use **ssh** and **scp** for the remote shell and copy commands. For more information, see “Installing and configuring OpenSSH” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Carefully consider the remote execution and remote copy tooling you want to use within your cluster. Once a cluster has been created, it is complicated to change, especially if additional nodes are added. The default tools as specified under **-r RemoteShellCommand** and **-R RemoteFileCopyCommand** by default use `/usr/bin/ssh` and `/usr/bin/scp` respectively. For more information, see “GPFS cluster creation considerations” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Parameters

-N NodeDesc[,NodeDesc...] | NodeFile

Specifies node descriptors, which provide information about nodes to be added to the cluster.

NodeFile

Specifies a file containing a list of node descriptors, one per line, to be added to the cluster.

NodeDesc[,NodeDesc...]

Specifies the list of nodes and node designations to be added to the GPFS cluster. Node descriptors are defined as:

mmcrcluster

NodeName:NodeDesignations:AdminNodeName

where:

NodeName

Specifies the host name or IP address of the node for GPFS daemon-to-daemon communication. For hosts with multiple adapters, see the *IBM Spectrum Scale: Advanced Administration Guide* and search on *Using remote access with public and private IP addresses*.

The host name or IP address must refer to the communication adapter over which the GPFS daemons communicate. Aliased interfaces are not allowed. Use the original address or a name that is resolved by the **host** command to that original address. You can specify a node using any of these forms:

- Short host name (for example, h135n01)
- Long, fully-qualified, host name (for example, h135n01.ibm.com)
- IP address (for example, 7.111.12.102). IPv6 addresses must be enclosed in brackets (for example, [2001:192::192:168:115:124]).

Regardless of which form you use, GPFS will resolve the input to a host name and an IP address and will store these in its configuration files. It is expected that those values will not change while the node belongs to the cluster.

NodeDesignations

An optional, "-" separated list of node roles:

- **manager** | **client** – Indicates whether a node is part of the node pool from which file system managers and token managers can be selected. The default is **client**.
- **quorum** | **nonquorum** – Indicates whether a node is counted as a quorum node. The default is **nonquorum**.

AdminNodeName

Specifies an optional field that consists of a node name to be used by the administration commands to communicate between nodes. If *AdminNodeName* is not specified, the *NodeName* value is used.

You must provide a *NodeDesc* for each node to be added to the GPFS cluster.

--ccr-enable

Enables the configuration server repository (CCR), which stores redundant copies of configuration data files on all quorum nodes. All GPFS administration commands, as well as file system mounts and daemon startups, work normally as long as a majority of quorum nodes are accessible. This is the default.

The CCR operation requires the use of the GSKit toolkit for authenticating network connections. As such, the **gpfs.gskit** package, which is available on all Editions, should be installed.

--ccr-disable

Indicates that the traditional primary/backup server-based configuration repository (non-CCR, earlier than GPFS 4.1) is to be used.

When using this option you must also specify a primary configuration server (**-p** option). It is suggested that you also specify a secondary GPFS cluster configuration server (**-s** option) to prevent the loss of configuration data in the event your primary GPFS cluster configuration server goes down. When the GPFS daemon starts up, at least one of the two GPFS cluster configuration servers must be accessible.

If your primary GPFS cluster configuration server fails and you have not designated a secondary server, the GPFS cluster configuration files are inaccessible, and any GPFS administration commands that are issued fail. File system mounts or daemon startups also fail if no GPFS cluster configuration server is available.

You are strongly advised to designate the cluster configuration servers as quorum nodes.

-p *PrimaryServer*

Specifies the primary GPFS cluster configuration server node used to store the GPFS configuration data. This node must be a member of the GPFS cluster. This option is necessary only when **-ccr-disable** is specified.

-s *SecondaryServer*

Specifies the secondary GPFS cluster configuration server node used to store the GPFS cluster data. This node must be a member of the GPFS cluster. This option is necessary only when **-ccr-disable** is specified.

-r *RemoteShellCommand*

Specifies the fully-qualified path name for the remote shell program to be used by GPFS. The default value is **/usr/bin/ssh**.

The remote shell command must adhere to the same syntax format as the **ssh** command, but may implement an alternate authentication mechanism.

-R *RemoteFileCopy*

Specifies the fully-qualified path name for the remote file copy program to be used by GPFS. The default value is **/usr/bin/scp**.

The remote copy command must adhere to the same syntax format as the **scp** command, but may implement an alternate authentication mechanism.

--use-sudo-wrapper

Specifies that the nodes in the cluster call the **ssh** sudo wrapper script and the **scp** sudo wrapper script as the remote shell program and the remote copy program. For more information see “Running IBM Spectrum Scale without remote root login” on page 21.

-C *ClusterName*

Specifies a name for the cluster. If the user-provided name contains dots, it is assumed to be a fully qualified domain name. Otherwise, to make the cluster name unique, the domain of the primary configuration server will be appended to the user-provided name.

If the **-C** flag is omitted, the cluster name defaults to the name of the primary GPFS cluster configuration server.

-U *DomainName*

Specifies the UID domain name for the cluster.

-A Specifies that GPFS daemons are to be automatically started when nodes come up. The default is not to start daemons automatically.

-c *ConfigFile*

Specifies a file containing GPFS configuration parameters with values different than the documented defaults. A sample file can be found in **/usr/lpp/mmfs/samples/mmfs.cfg.sample**. See the **mmchconfig** command for a detailed description of the different configuration parameters.

The **-c ConfigFile** parameter should be used only by experienced administrators. Use this file to set up only those parameters that appear in the **mmfs.cfg.sample** file. Changes to any other values may be ignored by GPFS. When in doubt, use the **mmchconfig** command instead.

--profile *ProfileName*

Specifies a predefined profile of attributes to be applied. System-defined profiles are located in **/usr/lpp/mmfs/profiles/**. All the configuration attributes listed under a cluster stanza will be changed as a result of this command.

The following system-defined profile names are accepted:

- **gpfsProtocolDefaults**
- **gpfsProtocolRandomIO**

mmcrcluster

A user's profiles must be installed in `/var/mmfs/etc/`. The profile file specifies GPFS configuration parameters with values different than the documented defaults. A user-defined profile must not begin with the string 'gpfs' and must have the `.profile` suffix.

User-defined profiles consist of the following stanzas:

```
%cluster:
[CommaSeparatedNodesOrNodeClasses:]ClusterConfigurationAttribute=Value
...
%filesystem:
FilesystemConfigurationAttribute=Value
```

See the **mmchconfig** command for a detailed description of the different configuration parameters. A sample file can be found in `/usr/lpp/mmfs/samples/sample.profile`.

Note: User-defined profiles should be used only by experienced administrators. When in doubt, use the **mmchconfig** command instead.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrcluster** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To create a GPFS cluster made of all of the nodes listed in the file `/u/admin/nodelist`, using node **k164n05** as the primary server, and node **k164n04** as the secondary server, issue:

```
mmcrcluster -N /u/admin/nodelist -p k164n05 -s k164n04
```

where `/u/admin/nodelist` has these contents:

```
k164n04.kgn.ibm.com:quorum
k164n05.kgn.ibm.com:quorum
k164n06.kgn.ibm.com
```

The output of the command is similar to:

```
Mon May 10 10:59:09 EDT 2010: mmcrcluster:
  Processing node k164n04.kgn.ibm.com
Mon May 10 10:59:09 EDT 2010: mmcrcluster:
  Processing node k164n05.kgn.ibm.com
Mon May 10 10:59:09 EDT 2010: mmcrcluster:
  Processing node k164n06.kgn.ibm.com
mmcrcluster: Command successfully completed
mmcrcluster: Warning: Not all nodes have proper
  GPFS license designations.
  Use the mmchlicense command to designate
  licenses as needed.
```

To confirm the creation, issue this command:

```
mmfsccluster
```

The system displays information similar to:

GPFS cluster information

=====

```
GPFS cluster name:      k164n05.kgn.ibm.com
GPFS cluster id:       680681562214606028
GPFS UID domain:      k164n05.kgn.ibm.com
Remote shell command: /usr/bin/ssh
Remote file copy command: /usr/bin/scp
```

GPFS cluster configuration servers:

```
Primary server:  k164n05.kgn.ibm.com
Secondary server: k164n04.kgn.ibm.com
```

Node Daemon node name IP address Admin node name Designation

```
1 k164n04.kgn.ibm.com 198.117.68.68 k164n04.kgn.ibm.com quorum
2 k164n05.kgn.ibm.com 198.117.68.71 k164n05.kgn.ibm.com quorum
3 k164n06.kgn.ibm.com 198.117.68.70 k164n06.kgn.ibm.com
```

See also

- “mmaddnode command” on page 245
- “mmchconfig command” on page 331
- “mmdelnode command” on page 460
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526

Location

/usr/lpp/mmfs/bin

mmcrfileset command

Creates a GPFS fileset.

Synopsis

```
mmcrfileset Device FilesetName [-p afmAttribute=Value...] [-t Comment]
  [--inode-space {new [--inode-limit MaxNumInodes[:NumInodesToPreallocate]] | ExistingFileset}]
  [--allow-permission-change PermissionChangeMode]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmcrfileset** command constructs a new fileset with the specified name. The new fileset is empty except for a root directory, and does not appear in the directory namespace until the **mmmlinkfileset** command is issued. The **mmcrfileset** command is separate from the **mmmlinkfileset** command to allow the administrator to establish policies and quotas on the fileset before it is linked into the namespace.

For information on GPFS filesets, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system to contain the new fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

FilesetName

Specifies the name of the newly created fileset.

-p *afmAttribute=Value*

Specifies an AFM configuration parameter and its value. More than one **-p** option can be specified.

The following AFM configuration parameter is required for the **mmcrfileset** command:

afmTarget

Identifies the home that is associated with the cache; specified in either of the following forms:

Protocol://[Host|Map]/Path

or

{Host|Map}:Path

where:

Protocol://

Specifies the transport protocol. Valid values are **nfs://** or **gpfs://**.

Host|Map

Host

Specifies the server domain name system (DNS) name or IP address.

Map

Specifies the export map name.

Notes:

1. When specifying **nfs://** as the value for *Protocol://*, you must provide a value for *Host* or *Map*.

- When specifying **gpfs://** as the value for *Protocol://*, do not provide a value for *Host*. However, provide a value for *Map* if it refers to an export map entry.

Path

Specifies the export path.

For example:

- The following command creates a single-writer AFM fileset in a GPFS file system fs3 with a remote file system mounted at /gpfs/theofs1 from node c41bn3, using protocol **nfs://**:

```
mmcrfileset fs3 singleWriter2 -p afmtarget=nfs://c41bn3/gpfs/theofs1/target2 -p afmnode=sw --inode-space new
Fileset singleWriter2 created with id 23 root inode 3145731.
```

- The following command creates a single-writer AFM fileset in a GPFS file system fs3 with a GPFS remote file system mounted at /gpfs/theofs1, using protocol **gpfs://**:

```
mmcrfileset fs3 singleWriter1 -p afmtarget=gpfs:///gpfs/theofs1/target1 -p afmnode=sw --inode-space new
Fileset singleWriter1 created with id 21 root inode 2883587.
```

Note that in this case **///** is needed because *Host* is not provided.

The following optional AFM configuration parameters are also valid:

afmAsyncDelay

Specifies (in seconds) the amount of time by which write operations are delayed (because write operations are asynchronous with respect to remote clusters). For write-intensive applications that keep writing to the same set of files, this delay is helpful because it replaces multiple writes to the home cluster with a single write containing the latest data. However, setting a very high value weakens the consistency of data on the remote cluster.

This configuration parameter is applicable only for writer caches (SW and IW), where data from cache is pushed to home.

Valid values are 1 through 2147483647. The default is 15.

afmDirLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a directory, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of that directory has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 60. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmDirOpenRefreshInterval

Controls the frequency of data revalidations that are triggered by such I/O operations as **read** or **write** (specified in seconds). After a directory has been cached, **open** requests resulting from I/O operations on that object are directed to the cached directory until the specified amount of time has passed. Once the specified amount of time has passed, the **open** request gets directed to a gateway node rather than to the cached directory.

Valid values are 0 through 2147483647. The default is 60. Setting a lower value guarantees a higher level of consistency.

afmEnableAutoEviction

Enables eviction on a given fileset. A **yes** value specifies that eviction is allowed on the fileset. A **no** value specifies that eviction is not allowed on the fileset.

See also the topic about cache eviction in the *IBM Spectrum Scale: Advanced Administration Guide*.

afmExpirationTimeout

Is used with **afmDisconnectTimeout** (which can be set only through **mmchconfig**) to control how long a network outage between the cache and home clusters can continue before the data in the cache is considered out of sync with home. After **afmDisconnectTimeout** expires, cached data

mmcrfileset

remains available until **afmExpirationTimeout** expires, at which point the cached data is considered expired and cannot be read until a reconnect occurs.

Valid values are 0 through 2147483647. The default is 300.

afmFileLookupRefreshInterval

Controls the frequency of data revalidations that are triggered by such lookup operations as **ls** or **stat** (specified in seconds). When a lookup operation is done on a file, if the specified amount of time has passed, AFM sends a message to the home cluster to find out whether the metadata of the file has been modified since the last time it was checked. If the time interval has not passed, AFM does not check the home cluster for updates to the metadata.

Valid values are 0 through 2147483647. The default is 30. In situations where home cluster data changes frequently, a value of 0 is recommended.

afmMode

Specifies the mode in which the cache operates. Valid values are the following:

single-writer | sw

Specifies single-writer mode.

read-only | ro

Specifies read-only mode. (For **mmcrfileset**, this is the default value.)

local-updates | lu

Specifies local-updates mode.

independent-writer | iw

Specifies independent-writer mode.

Primary

Specifies the primary mode for AFM asynchronous data replication.

Secondary

Specifies the secondary mode for AFM asynchronous data replication.

Changing from single-writer/read-only modes to read-only/local-updates/single-writer is supported. When changing from read-only to single-writer, the read-only cache is up-to-date. When changing from single-writer to read-only, all requests from cache should have been played at home. Changing from local-updates to read-only/local-updates/single-writer is restricted. A typical dataset is set up to include a single cache cluster in single-writer mode (which generates the data) and one or more cache clusters in local-updates or read-only mode. AFM single-writer/independent-writer filesets can be converted to primary using the conversion process described in the chapter about AFM disaster recovery in the *IBM Spectrum Scale: Advanced Administration Guide*. Primary/secondary filesets cannot be converted to AFM filesets.

In case of AFM asynchronous data replication, the **mmchfileset** command cannot be used to convert to primary from secondary. For information about converting to primary and secondary, see the chapter about AFM disaster recovery in the *IBM Spectrum Scale: Advanced Administration Guide*.

For more information, see the topic about caching modes in the *IBM Spectrum Scale: Advanced Administration Guide* chapter about active file management.

afmNumFlushThreads

Defines the number of threads used on each gateway to synchronize updates to the home cluster. The default value is 4, which is sufficient for most installations. The current maximum value is 1024, which is too high for most installations; setting this parameter to such an extreme value should be avoided.

afmParallelReadChunkSize

Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel reads. Values are interpreted in terms of bytes. The default value of this

parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelReadThreshold

Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default value is 1024 MB. The valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelWriteChunkSize

Defines the minimum chunk size of the write that needs to be distributed among the gateway nodes during parallel writes. Values are interpreted in terms of bytes. The default value of this parameter is 128 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmParallelWriteThreshold

Defines the threshold beyond which parallel writes become effective. Writes are split into chunks when file size exceeds this threshold value. Values are interpreted in terms of MB. The default value of this parameter is 1024 MB, and the valid range of values is 0 to 2147483647. It can be changed cluster wide with the **mmchconfig** command. It can be set at fileset level using **mmcrfileset** or **mmchfileset** commands.

afmPrefetchThreshold

Controls partial file caching and prefetching. Valid values are the following:

0 Enables full file prefetching. This is useful for sequentially accessed files that are read in their entirety, such as image files, home directories, and development environments. The file will be prefetched after three blocks have been read into the cache.

1-99

Specifies the percentage of file size that must be cached before the entire file is prefetched. A large value is suitable for a file accessed either randomly or sequentially but partially, for which it might be useful to ingest the rest of the file when most of it has been accessed.

100

Disables full file prefetching. This value only fetches and caches data that is read by the application. This is useful for large random-access files, such as databases, that are either too big to fit in the cache or are never expected to be read in their entirety. When all data blocks are accessed in the cache, the file is marked as cached.

0 is the default value.

For local-updates mode, the whole file is prefetched when the first update is made.

afmPrimaryId

Specifies the unique primary ID of the primary fileset for asynchronous data replication. This is used for connecting a secondary to a primary.

afmRPO

Specifies the recovery point objective (RPO) interval in minutes for a primary fileset.

afmShowHomeSnapshot

Controls the visibility of the home snapshot directory in cache. For this to be visible in cache, this variable has to be set to **yes**, and the snapshot directory name in cache and home should not be the same.

yes

Specifies that the home snapshot link directory is visible.

mmcrfileset

no Specifies that the home snapshot link directory is not visible.

See also the topic about peer snapshots in the *IBM Spectrum Scale: Advanced Administration Guide*.

-t *Comment*

Specifies an optional comment that appears in the output of the **mmlsfileset** command. This comment must be less than 256 characters in length.

--inode-space {**new** | *ExistingFileset*}

Specifies the type of fileset to create, which controls how inodes are allocated:

new

Creates an independent fileset and its own dedicated inode space.

ExistingFileset

Creates a dependent fileset that will share inode space with the specified *ExistingFileset*. The *ExistingFileset* can be **root** or any other independent fileset.

If **--inode-space** is not specified, a dependent fileset will be created in the root inode space.

--inode-limit *MaxNumInodes* [:*NumInodesToPreallocate*]

Specifies the inode limit for the new inode space. The *NumInodesToPreallocate* specifies an optional number of inodes to preallocate when the fileset is created. This option is valid only when creating an independent fileset with the **--inode-space new** parameter.

--allow-permission-change *PermissionChangeMode*

Specifies the new permission change mode. This mode controls how **chmod** and ACL operations are handled on objects in the fileset. Valid modes are as follows:

chmodOnly

Specifies that only the UNIX change mode operation (**chmod**) is allowed to change access permissions (ACL commands and API will not be accepted).

setAc1Only

Specifies that permissions can be changed using ACL commands and API only (**chmod** will not be accepted).

chmodAndSetAc1

Specifies that **chmod** and ACL operations are permitted. If the **chmod** command (or **setattr** file operation) is issued, the result depends on the type of ACL that was previously controlling access to the object:

- If the object had a Posix ACL, it will be modified accordingly.
- If the object had an NFSv4 ACL, it will be replaced by the given UNIX mode bits.

Note: This is the default setting when a fileset is created.

chmodAndUpdateAc1

Specifies that **chmod** and ACL operations are permitted. If **chmod** is issued, the ACL will be updated by privileges derived from UNIX mode bits.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrfileset** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This example creates a fileset in file system **gpfs1**:

```
mmcrfileset gpfs1 fset1
```

The system displays output similar to:

```
Fileset fset1 created with id 1.
```

2. This example adds **fset2** in file system **gpfs1** with the comment "another fileset":

```
mmcrfileset gpfs1 fset2 -t "another fileset"
```

The system displays output similar to:

```
Fileset fset2 created with id 2.
```

To confirm the change, issue this command:

```
mmlsfileset gpfs1 -L
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name Id RootInode ParentId Created      InodeSpace MaxInodes AllocInodes Comment
root  0      3      -- Mon Apr 12 16:31:05 2010      0           8001536   8001536   root fileset
fset1 1    13568      0 Mon Apr 12 16:32:28 2010      0             0           0
fset2 2    13569      0 Mon Apr 12 16:32:28 2010      0             0           0 another fileset
```

See also

- “mmchfileset command” on page 365
- “mmdelfileset command” on page 455
- “mmlinkfileset command” on page 517
- “mmlsfileset command” on page 532
- “mmunlinkfileset command” on page 687

Location

```
/usr/lpp/mmfs/bin
```

mmcrfs command

Creates a GPFS file system.

Synopsis

```
mmcrfs Device {"DiskDesc[;DiskDesc...]" | -F StanzaFile}
  [-A {yes | no | automount}] [-B BlockSize] [-D {posix | nfs4}]
  [-E {yes | no}] [-i InodeSize] [-j {cluster | scatter}]
  [-k {posix | nfs4 | all}] [-K {no | whenpossible | always}]
  [-L LogFileSize] [-m DefaultMetadataReplicas]
  [-M MaxMetadataReplicas] [-n NumNodes] [-Q {yes | no}]
  [-r DefaultDataReplicas] [-R MaxDataReplicas]
  [-S {yes | no | relatime}] [-T Mountpoint] [-t DriveLetter]
  [-v {yes | no}] [-z {yes | no}] [--filesetdf | --nofilesetdf]
  [--inode-limit MaxNumInodes[:NumInodesToPreallocate]]
  [--log-replicas LogReplicas] [--metadata-block-size MetadataBlockSize]
  [--perfileset-quota | --noperfileset-quota]
  [--mount-priority Priority] [--version VersionString]
  [--write-cache-threshold HAWCThreshold]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmcrfs** command to create a GPFS file system. The first two parameters *must* be *Device* and either *DiskDescList* or *StanzaFile* and they *must* be in that order. The block size and replication factors chosen affect file system performance. A maximum of 256 file systems can be mounted in a GPFS cluster at one time, including remote file systems.

When deciding on the maximum number of files (number of inodes) in a file system, consider that for file systems that will be doing parallel file creates, if the total number of free inodes is not greater than 5% of the total number of inodes, there is the potential for slowdown in file system access. The total number of inodes can be increased using the **mmchfs** command.

When deciding on a block size for a file system, consider these points:

- Supported block sizes are 64 KiB, 128 KiB, 256 KiB, 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, and 16 MiB.
- The GPFS block size determines:
 - The minimum disk space allocation unit. The minimum amount of space that file data can occupy is a sub-block. A sub-block is 1/32 of the block size.
 - The maximum size of a read or write request that GPFS sends to the underlying disk driver.
- From a performance perspective, it is recommended that you set the GPFS block size to match the application buffer size, the RAID stripe size, or a multiple of the RAID stripe size. If the GPFS block size does not match the RAID stripe size, performance may be severely degraded, especially for write operations. If GPFS Native RAID is in use, the block size must equal the vdisk track size. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.
- In file systems with a high degree of variance in the size of files within the file system, using a small block size would have a large impact on performance when accessing large files. In this kind of system it is suggested that you use a block size of 256 KB (8 KB sub-block). Even if only 1% of the files are large, the amount of space taken by the large files usually dominates the amount of space used on disk, and the waste in the sub-block used for small files is usually insignificant. For further performance information, see the GPFS white papers in the Techdocs Library (www.ibm.com/support/techdocs/atmastr.nsf/Web/WhitePapers).
- The effect of block size on file system performance largely depends on the application I/O pattern.
 - A larger block size is often beneficial for large sequential read and write workloads.

- A smaller block size is likely to offer better performance for small file, small random read and write, and metadata-intensive workloads.
6. The efficiency of many algorithms that rely on caching file data in a GPFS page pool depends more on the number of blocks cached rather than the absolute amount of data. For a page pool of a given size, a larger file system block size would mean fewer blocks cached. Therefore, when you create file systems with a block size larger than the default of 256 KB, it is recommended that you increase the page pool size in proportion to the block size.
 7. The file system block size must not exceed the value of the GPFS maximum file system block size. The default maximum block size is 1 MiB. If a larger block size is desired, use the **mmchconfig** command to increase the **maxblocksize** configuration parameter before starting GPFS.

Results

Upon successful completion of the **mmcrfs** command, these tasks are completed on all GPFS nodes:

- Mount point directory is created.
- File system is formatted.

Prior to GPFS 3.5, the disk information was specified in the form of disk descriptors defined as follows (with the second, third, and sixth fields reserved):

```
DiskName:::DiskUsage:FailureGroup:::StoragePool:
```

For backward compatibility, the **mmcrfs** command will still accept the traditional disk descriptors, but their use is discouraged.

Parameters

Device

The device name of the file system to be created.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**. However, file system names must be unique within a GPFS cluster. Do not specify an existing entry in **/dev**.

This must be the first parameter.

"DiskDesc[;DiskDesc...]"

A descriptor for each disk to be included. Each descriptor is separated by a semicolon (;). The entire list must be enclosed in quotation marks (' or "). The use of disk descriptors is discouraged.

-F StanzaFile

Specifies a file containing the NSD stanzas and pool stanzas for the disks to be added to the file system.

NSD stanzas have this format:

```
%nsd:
  nsd=NsdName
  usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}
  failureGroup=FailureGroup
  pool=StoragePool
  servers=ServerList
  device=DiskName
```

where:

nsd=NsdName

The name of an NSD previously created by the **mmcrnsd** command. For a list of available disks, issue the **mmlsnsd -F** command. This clause is mandatory for the **mmcrfs** command.

usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}

Specifies the type of data to be stored on the disk:

mmcrfs

dataAndMetadata

Indicates that the disk contains both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disk contains data and does not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disk contains metadata and does not contain data.

descOnly

Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain disaster-recovery configurations. For more information, see the *IBM Spectrum Scale: Advanced Administration Guide* and search for "Synchronous mirroring utilizing GPFS replication"

failureGroup=FailureGroup

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

pool=StoragePool

Specifies the storage pool to which the disk is to be assigned. If this name is not provided, the default is **system**.

Only the system storage pool can contain **metadataOnly**, **dataAndMetadata**, or **descOnly** disks. Disks in other storage pools must be **dataOnly**.

servers=ServerList

A comma-separated list of NSD server nodes. This clause is ignored by the **mmcrfs** command.

device=DiskName

The block device name of the underlying disk device. This clause is ignored by the **mmcrfs** command.

Pool stanzas have this format:

```
%pool:  
  pool=StoragePoolName  
  blockSize=BlockSize  
  usage={dataOnly | metadataOnly | dataAndMetadata}  
  layoutMap={scatter | cluster}  
  allowWriteAffinity={yes | no}  
  writeAffinityDepth={0 | 1 | 2}  
  blockGroupFactor=BlockGroupFactor
```

where:

pool=StoragePoolName

Is the name of a storage pool.

blockSize=BlockSize

Specifies the block size of the disks in the storage pool.

usage={dataOnly | metadataOnly | dataAndMetadata}

Specifies the type of data to be stored in the storage pool:

dataAndMetadata

Indicates that the disks in the storage pool contain both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disks contain data and do not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disks contain metadata and do not contain data.

layoutMap={scatter | cluster}

Specifies the block allocation map type. When allocating blocks for a given file, GPFS first uses a round-robin algorithm to spread the data across all disks in the storage pool. After a disk is selected, the location of the data block on the disk is determined by the block allocation map type. If **cluster** is specified, GPFS attempts to allocate blocks in clusters. Blocks that belong to a particular file are kept adjacent to each other within each cluster. If **scatter** is specified, the location of the block is chosen randomly.

The **cluster** allocation method may provide better disk performance for some disk subsystems in relatively small installations. The benefits of clustered block allocation diminish when the number of nodes in the cluster or the number of disks in a file system increases, or when the file system's free space becomes fragmented. The **cluster** allocation method is the default for GPFS clusters with eight or fewer nodes and for file systems with eight or fewer disks.

The **scatter** allocation method provides more consistent file system performance by averaging out performance variations due to block location (for many disk subsystems, the location of the data relative to the disk edge has a substantial effect on performance). This allocation method is appropriate in most cases and is the default for GPFS clusters with more than eight nodes or file systems with more than eight disks.

The block allocation map type cannot be changed after the storage pool has been created.

allowWriteAffinity={yes | no}

Indicates whether the GPFS File Placement Optimizer (FPO) feature is to be enabled for the storage pool. See the section about the GPFS File Placement Optimizer in the *IBM Spectrum Scale: Advanced Administration Guide* for additional information.

writeAffinityDepth={0 | 1 | 2}

Specifies the allocation policy to be used by the node writing the data.

A write affinity depth of 0 indicates that each replica is to be striped across the disks in a cyclical fashion with the restriction that no two disks are in the same failure group. By default, the unit of striping is a block; however, if the block group factor is specified in order to exploit chunks, the unit of striping is a chunk.

A write affinity depth of 1 indicates that the first copy is written to the writer node. The second copy is written to a different rack. The third copy is written to the same rack as the second copy, but on a different half (which can be composed of several nodes).

A write affinity depth of 2 indicates that the first copy is written to the writer node. The second copy is written to the same rack as the first copy, but on a different half (which can be composed of several nodes). The target node is determined by a hash value on the fileset ID of the file, or it is chosen randomly if the file does not belong to any fileset. The third copy is striped across the disks in a cyclical fashion with the restriction that no two disks are in the same failure group.

mmcrfs

This behavior can be altered on an individual file basis by using the **--write-affinity-failure-group** option of the **mmchattr** command.

This parameter is ignored if write affinity is disabled for the storage pool.

blockGroupFactor=BlockGroupFactor

Specifies how many file system blocks are laid out sequentially on disk to behave like a single large block. This option only works if **--allow-write-affinity** is set for the data pool. This applies only to a new data block layout; it does not migrate previously existing data blocks.

See the section about File Placement Optimizer in the *IBM Spectrum Scale: Advanced Administration Guide*.

-A {yes | no | automount}

Indicates when the file system is to be mounted:

yes

When the GPFS daemon starts. This is the default.

no Manual mount.

automount

On non-Windows nodes, when the file system is first accessed. On Windows nodes, when the GPFS daemon starts.

-B BlockSize

Specifies the size of data blocks. Must be 64 KiB, 128 KiB, 256 KiB (the default), 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, or 16 MiB. Specify this value with the character **K** or **M**, for example **512K**.

-D {nfs4 | posix}

Specifies whether a deny-write open lock will block writes, which is expected and required by NFS V4. File systems supporting NFS V4 must have **-D nfs4** set. The option **-D posix** allows NFS writes even in the presence of a deny-write open lock. If you intend to export the file system using NFS V4 or Samba, you must use **-D nfs4**. For NFS V3 (or if the file system is not NFS exported at all) use **-D posix**. The default is **-D nfs4**.

-E {yes | no}

Specifies whether to report *exact* **mtime** values (**-E yes**), or to periodically update the **mtime** value for a file system (**-E no**). If it is more desirable to display exact modification times for a file system, specify or use the default **-E yes**.

-i InodeSize

Specifies the byte size of inodes. Supported inode sizes are 512, 1024, and 4096 bytes. The default is 4096.

-j {cluster | scatter}

Specifies the default block allocation map type to be used if **layoutMap** is not specified for a given storage pool.

-k {posix | nfs4 | all}

Specifies the type of authorization supported by the file system:

posix

Traditional GPFS ACLs only (NFS V4 and Windows ACLs are not allowed). Authorization controls are unchanged from earlier releases.

nfs4

Support for NFS V4 and Windows ACLs only. Users are not allowed to assign traditional GPFS ACLs to any file system objects (directories and individual files).

all

Any supported ACL type is permitted. This includes traditional GPFS (**posix**) and NFS V4 NFS V4 and Windows ACLs (**nfs4**).

The administrator is allowing a mixture of ACL types. For example, **fileA** may have a **posix** ACL, while **fileB** in the same file system may have an NFS V4 ACL, implying different access characteristics for each file depending on the ACL type that is currently assigned. The default is **-k all**.

Avoid specifying **nfs4** or **all** unless files will be exported to NFS V4 or Samba clients, or the file system will be mounted on Windows. NFS V4 and Windows ACLs affect file attributes (mode) and have access and authorization characteristics that are different from traditional GPFS ACLs.

-K {no | whenpossible | always}

Specifies whether strict replication is to be enforced:

no Indicates that strict replication is not enforced. GPFS will try to create the needed number of replicas, but will still return EOK as long as it can allocate at least one replica.

whenpossible

Indicates that strict replication is enforced provided the disk configuration allows it. If the number of failure groups is insufficient, strict replication will not be enforced. This is the default value.

always

Indicates that strict replication is enforced.

For more information, see the topic "Strict replication" in the *IBM Spectrum Scale: Problem Determination Guide*.

-L *LogFileSize*

Specifies the size of the internal log files. The *LogFileSize* specified must be a multiple of the metadata block size. The default size is 4 MB or the metadata block size, whichever is larger. The minimum size is 256 KB and the maximum size is 1024 MB. Specify this value with the K or M character, for example: 8M.

In most cases, allowing the log file size to default works well. An increased log file size is useful for file systems that have a large amount of metadata activity, such as creating and deleting many small files or performing extensive block allocation and deallocation of large files.

-m *DefaultMetadataReplicas*

Specifies the default number of copies of inodes, directories, and indirect blocks for a file. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of *MaxMetadataReplicas*. The default is 1.

-M *MaxMetadataReplicas*

Specifies the default maximum number of copies of inodes, directories, and indirect blocks for a file. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be less than the value of *DefaultMetadataReplicas*. The default is 2.

-n *NumNodes*

The estimated number of nodes that will mount the file system in the local cluster and all remote clusters. This is used as a best guess for the initial size of some file system data structures. The default is 32. This value can be changed after the file system has been created.

When you create a GPFS file system, you might want to overestimate the number of nodes that will mount the file system. GPFS uses this information for creating data structures that are essential for achieving maximum parallelism in file system operations (For more information, see *GPFS architecture in IBM Spectrum Scale: Concepts, Planning, and Installation Guide*). If you are sure there will never be more than 64 nodes, allow the default value to be applied. If you are planning to add nodes to your system, you should specify a number larger than the default.

mmcrfs

-Q {yes | no}

Activates quotas automatically when the file system is mounted. The default is **-Q no**. Issue the **mmdefedquota** command to establish default quota values. Issue the **mmedquota** command to establish explicit quota values.

To activate GPFS quota management after the file system has been created:

1. Mount the file system.
2. To establish default quotas:
 - a. Issue the **mmdefedquota** command to establish default quota values.
 - b. Issue the **mmdefquotaon** command to activate default quotas.
3. To activate explicit quotas:
 - a. Issue the **mmedquota** command to activate quota values.
 - b. Issue the **mmquotaon** command to activate quota enforcement.

-r DefaultDataReplicas

Specifies the default number of copies of each data block for a file. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be greater than the value of *MaxDataReplicas*. The default is 1.

-R MaxDataReplicas

Specifies the default maximum number of copies of data blocks for a file. Valid values are 1, 2, and (for GPFS V3.5.0.7 and later) 3. This value cannot be less than the value of *DefaultDataReplicas*. The default is 2.

-S {yes | no | relatime}

Suppresses the periodic updating of the value of **atime** as reported by the **gpfs_stat()**, **gpfs_fstat()**, **stat()**, and **fstat()** calls. The default value is **-S no**. Specifying **-S yes** for a new file system results in reporting the time the file system was created.

If **relatime** is specified, the file access time is updated only if the existing access time is older than the value of the **atimeDeferredSeconds** configuration attribute or the existing file modification time is greater than the existing access time.

-T MountPoint

Specifies the mount point directory of the GPFS file system. If it is not specified, the mount point will be set to *DefaultMountDir/Device*. The default value for *DefaultMountDir* is */gpfs* but, it can be changed with the **mmchconfig** command.

-t DriveLetter

Specifies the drive letter to use when the file system is mounted on Windows.

-v {yes | no}

Verifies that specified disks do not belong to an existing file system. The default is **-v yes**. Specify **-v no** only when you want to reuse disks that are no longer needed for an existing file system. If the command is interrupted for any reason, use **-v no** on the next invocation of the command.

Important: Using **-v no** on a disk that already belongs to a file system will corrupt that file system. This will not be noticed until the next time that file system is mounted.

-z {yes | no}

Enable or disable DMAPI on the file system. Turning this option on will require an external data management application such as Tivoli Storage Manager (TSM) hierarchical storage management (HSM) before the file system can be mounted. The default is **-z no**. For further information on DMAPI for GPFS, see the *IBM Spectrum Scale: Data Management API Guide*.

--filesetdf

Specifies that when quotas are enforced for a fileset, the numbers reported by the **df** command are based on the quotas for the fileset (rather than the entire file system). This option affects the **df** command behavior only on Linux nodes.

--nofilesetdf

Specifies that when quotas are enforced for a fileset, the numbers reported by the **df** command are based on the quotas for the entire file system (rather than individual filesets. This is the default.

--inode-limit *MaxNumInodes*[:*NumInodesToPreallocate*]

Specifies the maximum number of files in the file system.

For file systems on which you intend to create files in parallel, if the total number of free inodes is not greater than 5% of the total number of inodes, file system access might slow down. Take this into consideration when creating your file system.

The parameter *NumInodesToPreallocate* specifies the number of inodes that the system will immediately preallocate. If you do not specify a value for *NumInodesToPreallocate*, GPFS will dynamically allocate inodes as needed.

You can specify the *NumInodes* and *NumInodesToPreallocate* values with a suffix, for example 100K or 2M. Note that in order to optimize file system operations, the number of inodes that are actually created may be greater than the specified value.

--log-replicas *LogReplicas*

Specifies the number of recovery log replicas. Valid values are **1**, **2**, **3**, or **DEFAULT**. If not specified, or if **DEFAULT** is specified, the number of log replicas is the same as the number of metadata replicas currently in effect for the file system.

This option is only applicable if the recovery log is stored in the **system.log** storage pool.

--metadata-block-size *MetadataBlockSize*

Specifies the block size for the system storage pool, provided its usage is set to **metadataOnly**. Valid values are the same as those listed for **-B** *BlockSize*.

--perfileset-quota

Sets the scope of user and group quota limit checks to the individual fileset level (rather than the entire file system).

--noperfileset-quota

Sets the scope of user and group quota limit checks to the entire file system (rather than per individual fileset). This is the default.

--mount-priority *Priority*

Controls the order in which the individual file systems are mounted at daemon startup or when one of the **all** keywords is specified on the **mmmount** command.

File systems with higher *Priority* numbers are mounted after file systems with lower numbers. File systems that do not have mount priorities are mounted last. A value of zero indicates no priority. This is the default.

--version *VersionString*

Enable only the file system features that are compatible with the specified release. The lowest allowed *VersionString* is 3.1.0.0.

The default value is the current product version, which enables all currently available features but prevents nodes that are running earlier GPFS releases from accessing the file system. Windows nodes can mount only file systems that are created with GPFS 3.2.1.5 or later.

--profile *ProfileName*

Specifies a predefined profile of attributes to be applied. System-defined profiles are located in `/usr/lpp/mmfs/profiles/`. All the file system attributes listed under a file system stanza will be changed as a result of this command. The following system-defined profile names are accepted:

- `gpfsProtocolDefaults`
- `gpfsProtocolRandomIO`

The file system attributes will be applied at file system creation. If there is a current profile in place on the system (use `mmlsconfig` profile to check), then the file system will be created with those

mmcrfs

attributes and values listed in the profile's file system stanza. The default is to use whatever attributes and values associate with the current profile setting.

Furthermore, any and all file system attributes from an installed profile file can be by-passed with '--profile=userDefinedProfile', where the userDefinedProfile is a profile file has been installed by the user in /var/mmfs/etc/.

User-defined profiles consist of the following stanzas:

```
%cluster:
[CommaSeparatedNodesOrNodeClasses:]ClusterConfigurationAttribute=Value
...
%filesystem:
FilesystemConfigurationAttribute=Value
...
```

A sample file can be found in /usr/lpp/mmfs/samples/sample.profile. See the **mmchconfig** command for a detailed description of the different configuration parameters.

User-defined profiles should be used only by experienced administrators. When in doubt, use the **mmchconfig** command instead.

--write-cache-threshold *HAWCThreshold*

Specifies the maximum length (in bytes) of write requests that will be initially buffered in the highly-available write cache before being written back to primary storage. Only synchronous write requests are guaranteed to be buffered in this fashion.

A value of 0 disables this feature. 64K is the maximum supported value. Specify in multiples of 4K.

This feature can be enabled or disabled at any time (the file system does not need to be unmounted). For more information about this feature, see the topic *Highly-available write cache (HAWC)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

This example shows how to create a file system named **gpfs1** using three disks, each with a block size of 512 KB, allowing metadata and data replication to be 2, turning quotas on, and creating /gpfs1 as the mount point. The NSD stanzas describing the three disks are assumed to have been placed in file/tmp/freedisks. To complete this task, issue the command:

```
mmcrfs gpfs1 -F /tmp/freedisks -B 512K -m 2 -r 2 -Q yes -T /gpfs1
```

The system displays output similar to:

```
The following disks of gpfs1 will be formatted on node c21f1n13:
hd2n97: size 1951449088 KB
hd3n97: size 1951449088 KB
hd4n97: size 1951449088 KB
```

```
Formatting file system ...
Disks up to size 16 TB can be added to storage pool 'system'.
Creating Inode File
Creating Allocation Maps
Creating Log Files
Clearing Inode Allocation Map
Clearing Block Allocation Map
Formatting Allocation Map for storage pool 'system'
 19 % complete on Tue Feb 28 18:03:20 2012
 42 % complete on Tue Feb 28 18:03:25 2012
 62 % complete on Tue Feb 28 18:03:30 2012
 79 % complete on Tue Feb 28 18:03:35 2012
 96 % complete on Tue Feb 28 18:03:40 2012
100 % complete on Tue Feb 28 18:03:41 2012
Completed creation of file system /dev/gpfs1.
mmcrfs: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

See also

- “mmchfs command” on page 371
- “mmdelfs command” on page 458
- “mmdf command” on page 470
- “mmedquota command” on page 481
- “mmfsck command” on page 487
- “mmlsfs command” on page 536
- “mmlspool command” on page 554

Location

/usr/lpp/mmfs/bin

mmcrnodeclass command

Creates user-defined node classes.

Synopsis

```
mmcrnodeclass ClassName -N {Node[,Node...] | NodeFile | NodeClass}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmcrnodeclass** command to create user-defined node classes. After a node class is created, it can be specified as an argument on commands that accept the **-N *NodeClass*** option.

Parameters

ClassName

Specifies a name that uniquely identifies the user-defined node classes to create. An existing node name cannot be specified. Class names that end with **nodes** or system-defined node classes are reserved for use by GPFS.

-N {*Node*[,*Node*...] | *NodeFile* | *NodeClass*}

Specifies the nodes and node classes that will become members of the user-defined node class *ClassName*.

NodeClass cannot be a node class that already contains other node classes. For example, two user-defined node classes called **siteA** and **siteB** could be used to create a new node class called **siteAandB**, as follows:

```
mmcrnodeclass siteAandB -N siteA,siteB
```

The **siteAandB** node class cannot later be specified for *NodeClass* when creating new node classes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrnodeclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To create a user-defined node class called **siteA** that contains nodes **c8f2c4vp1** and **c8f2c4vp2**, issue this command:

```
mmcrnodeclass siteA -N c8f2c4vp1,c8f2c4vp2
```

The system displays information similar to:

```
mmcrnodeclass: Propagating the cluster configuration data to all
  affected nodes. This is an asynchronous process.
```

To display the member list of **siteA**, issue this command:

```
mmlsnodeclass siteA
```

The system displays information similar to:

Node Class Name	Members
siteA	c8f2c4vp1,c8f2c4vp2

See also

- “mmchnodeclass command” on page 385
- “mmdelnodeclass command” on page 463
- “mmlsnodeclass command” on page 546

Location

```
/usr/lpp/mmfs/bin
```

mmcrnsd command

Creates Network Shared Disks (NSDs) used by GPFS.

Synopsis

```
mmcrnsd -F StanzaFile [-v {yes | no}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmcrnsd** command is used to create cluster-wide names for NSDs used by GPFS.

This is the first GPFS step in preparing disks for use by a GPFS file system. The input to this command consists of a file containing NSD stanzas describing the properties of the disks to be created. This file may be updated as necessary by the **mmcrnsd** command and can be supplied as input to the **mmcrfs**, **mmadddisk**, or **mmrpldisk** command.

The names created by the **mmcrnsd** command are necessary since disks connected to multiple nodes may have different disk device names on each node. The NSD names uniquely identify each disk. This command must be run for all disks that are to be used in GPFS file systems. The **mmcrnsd** command is also used to assign each disk an NSD server list that can be used for I/O operations on behalf of nodes that do not have direct access to the disk.

To identify that a disk has been processed by the **mmcrnsd** command, a unique NSD volume ID is written on sector 2 of the disk. All of the NSD commands (**mmcrnsd**, **mmlsnsd**, and **mmdelnsd**) use this unique NSD volume ID to identify and process NSDs.

After the NSDs are created, the GPFS cluster data is updated and they are available for use by GPFS.

Note: It is customary to use whole LUNs as NSDs. This generally provides the best performance and fault isolation. When SCSI-3 PR is in use, whole LUN use is required. In other deployment scenarios, it is possible to use disk partitions rather than whole LUNs, as long as care is taken to ensure that sharing of the same LUN through multiple partitions does not have an undesirable performance impact.

On Windows, GPFS will only create NSDs from empty disk drives. **mmcrnsd** accepts Windows *Basic* disks or *Unknown/Not Initialized* disks. It always re-initializes these disks so that they become *Basic GPT Disks* with a single *GPFS partition*. NSD data is stored in GPFS partitions. This allows other operating system components to recognize the disks are used. **mmdelnsd** deletes the partition tables created by **mmcrnsd**.

Results

Upon successful completion of the **mmcrnsd** command, these tasks are completed:

- NSDs are created.
- The *StanzaFile* contains NSD names to be used as input to the **mmcrfs**, **mmadddisk**, or the **mmrpldisk** commands.
- A unique NSD volume ID to identify each disk as an NSD has been written on sector 2.
- An entry for each new disk is created in the GPFS cluster data.

Parameters

-F *StanzaFile*

Specifies the file containing the NSD stanzas for the disks to be created. NSD stanzas have this format:

```
%nsd: device=DiskName
      nsd=NsdName
      servers=ServerList
      usage={dataOnly | metadataOnly | dataAndMetadata | descOnly | localCache}
      failureGroup=FailureGroup
      pool=StoragePool
```

where:

device=*DiskName*

On UNIX, the block device name appearing in **/dev** for the disk you want to define as an NSD. Examples of disks that are accessible through a block device are SAN-attached disks. If server nodes are specified, *DiskName* must be the **/dev** name for the disk device of the first listed NSD server node.

On Windows, the disk number (for example, 3) of the disk you want to define as an NSD. Disk numbers appear in Windows Disk Management console and the DISKPART command line utility. If a server node is specified, *DiskName* must be the disk number from the first NSD server node defined in the server list.

For the latest supported disk types, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

This clause is mandatory for the **mmcrnsd** command.

nsd=*NsdName*

Specify the name you desire for the NSD to be created. This name must not already be used as another GPFS disk name, and it must not begin with the reserved string 'gpfs'.

Note: This name can contain only the following characters: 'A' through 'Z', 'a' through 'z', '0' through '9', or '_' (the underscore). All other characters are not valid.

If this clause is not specified, GPFS will generate a unique name for the disk and will add the appropriate **nsd**=*NsdName* clause to the stanza file. The NSD is assigned a name according to the convention:

gpfs*NN***nsd**

where *NN* is a unique nonnegative integer not used in any prior NSD.

servers=*ServerList*

Is a comma-separated list of NSD server nodes. You may specify up to eight NSD servers in this list. The defined NSD will preferentially use the first server on the list. If the first server is not available, the NSD will use the next available server on the list.

When specifying server nodes for your NSDs, the output of the **mmlscluster** command lists the host name and IP address combinations recognized by GPFS. The utilization of aliased host names not listed in the **mmlscluster** command output may produce undesired results.

There are two cases where a server list either must be omitted or is optional:

- For GPFS Native RAID, a server list is not allowed. The servers are determined from the underlying vdisk definition. For more information about GPFS Native RAID, see the *GPFS Native RAID: Administration and Programming Reference (SA23-1354)*.
- For SAN configurations where the disks are SAN-attached to all nodes in the cluster, a server list is optional. However, if all nodes in the cluster do not have access to the disk, or if the file system to which the disk belongs is to be accessed by other GPFS clusters, you must specify a server list.

mmcrnsd

usage={dataOnly | metadataOnly | dataAndMetadata | descOnly | localCache}

Specifies the type of data to be stored on the disk:

dataAndMetadata

Indicates that the disk contains both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disk contains data and does not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disk contains metadata and does not contain data.

descOnly

Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain disaster recovery configurations. For more information, see the help topic "Synchronous mirroring utilizing GPFS replication" in *IBM Spectrum Scale: Advanced Administration Guide*.

localCache

Indicates that the disk is to be used as a local read-only cache device.

This clause is ignored by the **mmcrnsd** command and is passed unchanged to the output file produced by the **mmcrnsd** command.

failureGroup=FailureGroup

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

This clause is ignored by the **mmcrnsd** command, and is passed unchanged to the output file produced by the **mmcrnsd** command.

pool=StoragePool

Specifies the name of the storage pool to which the NSD is assigned. This clause is ignored by the **mmcrnsd** command and is passed unchanged to the output file produced by the **mmcrnsd** command.

The default value for **pool** is **system**. In IBM Spectrum Scale Express Edition, **system** is the only value for pool that can be used, either by default or explicitly specified.

-v {yes | no}

Verify the disks are not already formatted as an NSD.

A value of **-v yes** specifies that the NSDs are to be created only if each disk has not been formatted by a previous invocation of the **mmcrnsd** command, as indicated by the NSD volume ID on sector 2 of the disk. A value of **-v no** specifies that the disks are to be created irrespective of their previous state. The default is **-v yes**.

Important: Using **-v no** when a disk already belongs to a file system can corrupt that file system by making that physical disk undiscoverable by that file system. This will not be noticed until the next time that file system is mounted.

Upon successful completion of the **mmcrnsd** command, the *StanzaFile* file is rewritten to reflect changes made by the command, as follows:

- If an NSD stanza is found to be in error, the stanza is commented out.
- If an **nsd=NsdName** clause is not specified, and an NSD name is generated by GPFS, an **nsd=** clause will be inserted in the corresponding stanza.

You must have **write** access to the directory where the *StanzaFile* file is located in order to rewrite the created NSD information.

The disk usage, failure group, and storage pool specifications are preserved only if you use the rewritten file produced by the **mmcrnsd** command. If you do not use this file, you must either accept the default values or specify new values when creating NSD stanzas for other commands.

Prior to GPFS 3.5, the disk information was specified in the form of disk descriptors defined as:

```
DiskName:ServerList::DiskUsage:FailureGroup:DesiredName:StoragePool
```

For backward compatibility, the **mmcrnsd** command will still accept the traditional disk descriptors, but their use is discouraged. For additional information about GPFS stanzas, see the help topic "Stanza files" in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrnsd** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To create two new NSDs from the stanza file `/tmp/newNSDstanza` containing:

```
%nsd: device=/dev/sdavl
      servers=k145n05,k145n06
      failureGroup=4
```

```
%nsd:
      device=/dev/sdav2
      nsd=sd2pA
      servers=k145n06,k145n05
      usage=dataOnly
      failureGroup=5
      pool=poolA
```

Issue this command:

```
mmcrnsd -F /tmp/newNSDstanza
```

mmcrnsd

The output is similar to this:

```
mmcrnsd: Processing disk sdav1
mmcrnsd: Processing disk sdav2
mmcrnsd: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

As a result, two NSDs will be created. The first one will be assigned a name by the system, for example, **gpfs1023nsd**. The second disk will be assigned the name **sd2pA** (as indicated by the **nsd=** clause in the stanza).

The `newNSDstanza` file will be rewritten and will look like this (note the addition of an **nsd=** clause in the first stanza):

```
%nsd: nsd=gpfs1023nsd device=/dev/sdav1
servers=k145n05,k145n06
failureGroup=4
```

```
%nsd:
device=/dev/sdav2
nsd=sd2pA
servers=k145n06,k145n05
usage=dataOnly
failureGroup=5
pool=poolA
```

See also

- “`mmadddisk` command” on page 239
- “`mmchnsd` command” on page 388
- “`mmcrfs` command” on page 414
- “`mmdeldisk` command” on page 449
- “`mmdeinsd` command” on page 465
- “`mmlsnsd` command” on page 549
- “`mmrpldisk` command” on page 648

Location

`/usr/lpp/mmfs/bin`

mmcrsnapshot command

Creates a snapshot of a file system or fileset at a single point in time.

Synopsis

```
mmcrsnapshot Device SnapshotName [-j Fileset]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmcrsnapshot** command to create global snapshots or fileset snapshots at a single point in time. System data and existing snapshots are not copied. The snapshot function allows a backup or mirror program to run concurrently with user updates and still obtain a consistent copy of the file system as of the time the copy was created. Snapshots also provide an online backup capability that allows easy recovery from common problems such as accidental deletion of a file, and comparison with older versions of a file.

A global snapshot is an exact copy of changed data in the active files and directories of a file system. Snapshots of a file system are read-only and they appear in a **.snapshots** directory located in the file system root directory. The files and attributes of the file system can be changed only in the active copy.

A fileset snapshot is an exact copy of changed data in the active files and directories of an independent fileset plus all dependent filesets. Fileset snapshots are read-only and they appear in a **.snapshots** directory located in the root directory of the fileset. The files and attributes of the fileset can be changed only in the active copy.

Snapshots may be deleted only by issuing the **mmdelsnapshot** command. The **.snapshots** directory cannot be deleted, though it can be renamed with the **mmsnapdir** command using the **-s** option.

Because global snapshots are not full, independent copies of the entire file system, they should not be used as protection against media failures. For protection against media failures, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on “recoverability considerations”.

For more information on global snapshots, see “Creating and maintaining snapshots of GPFS file systems” in the *IBM Spectrum Scale: Advanced Administration Guide*.

For more information on fileset snapshots, see “Fileset-level snapshots” in the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system for which the snapshot is to be created. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

This must be the first parameter.

SnapshotName

Specifies the name given to the snapshot.

For a global snapshot, *SnapshotName* appears as a subdirectory of the **.snapshots** directory in the root directory of the file system. Each global snapshot name must be unique from any other global snapshots. If you do not want to traverse the file system's root to access the global snapshot, a more convenient mechanism that enables a connection in each directory of the active file system can be enabled with the **-a** option of the **mmsnapdir** command.

mmcrsnapshot

Note: Ensure that the snapshot name is using the "@GMT-yyyy.MM.dd-HH.mm.ss" format in order to be identifiable by the Windows VSS.

For a fileset snapshot, *SnapshotName* appears as a subdirectory of the **.snapshots** directory in the root directory of the fileset. Fileset snapshot names can be duplicated across different filesets. A fileset snapshot can also have the same name as a global snapshot. The **mmsnapdir** command provides an option to make global snapshots also available through the **.snapshots** in the root directory of all independent filesets.

-j *Fileset*

Creates a snapshot that only includes the specified fileset plus all dependent filesets that share the same inode space. *Fileset* must refer to an independent fileset. If **-j** is not specified, the **mmcrsnapshot** command creates a global snapshot that includes all filesets.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmcrsnapshot** command when creating global snapshots.

Independent fileset owners can run the **mmcrsnapshot** command to create snapshots of the filesets they own.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To create a global snapshot **snap1**, for the file system **fs1**, issue this command:

```
mmcrsnapshot fs1 snap1
```

The output is similar to this:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
Snapshot snap1 created with id 1.
```

Before issuing the command, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3
```

After the command has been issued, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3
```

```
/fs1/.snapshots/snap1/file1
/fs1/.snapshots/snap1/userA/file2
/fs1/.snapshots/snap1/userA/file3
```

If a second snapshot were to be created at a later time, the first snapshot would remain as is. Snapshots are made only of active file systems, not existing snapshots. For example:

```
mmcrsnapshot fs1 snap2
```

The output is similar to this:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
Snapshot snap2 created with id 2.
```

After the command has been issued, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3

/fs1/.snapshots/snap1/file1
/fs1/.snapshots/snap1/userA/file2
/fs1/.snapshots/snap1/userA/file3

/fs1/.snapshots/snap2/file1
/fs1/.snapshots/snap2/userA/file2
/fs1/.snapshots/snap2/userA/file3
```

2. To create a snapshot **Snap3** of the fileset **FsetF5-V2**, for the file system **fs1**, issue this command:

```
mmcrsnapshot fs1 Snap3 -j FsetF5-V2
```

The system displays output similar to:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
Snapshot Snap3 created with id 69.
```

To display the snapshot that contains the **FsetF5-V2** fileset, issue this command:

```
mmlssnapshot fs1 -j FsetF5-V2
```

The system displays output similar to:

```
Snapshots in file system fs1:
Directory          SnapId   Status  Created                Fileset
Snap3              69      Valid  Wed Feb 1 12:55:51 2012 FsetF5-V2
```

3. To create a snapshot of the **gpfs0** file system that can be viewed over SMB protocol with Windows VSS, issue this command:

```
mmcrsnapshot gpfs0 $(date --utc +%GMT-%Y.%m.%d-%H.%M.%S)
```

The system displays output similar to:

```
mmcrsnapshot gpfs0 $(date --utc +%GMT-%Y.%m.%d-%H.%M.%S)
Flushing dirty data for snapshot @GMT-2015.10.02-21.03.33...
Quiescing all file system operations.
Snapshot @GMT-2015.10.02-21.03.33 created with id 7.
```

See also

- “mmdelsnapshot command” on page 467
- “mmlssnapshot command” on page 563
- “mmrestorefs command” on page 635
- “mmsnapdir command” on page 674

Location

```
/usr/lpp/mmfs/bin
```

mmdefedquota

mmdefedquota command

Sets default quota limits.

Synopsis

```
mmdefedquota {-u | -g | -j} Device
```

or

```
mmdefedquota {-u | -g} Device:Fileset
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmdefedquota** command to set or change default quota limits. Default quota limits can be set for new users, groups, and filesets for a specified file system. Default quota limits can also be applied at a more granular level for new users and groups in a specified fileset.

Default quota limits can be set or changed only if the **-Q yes** option is in effect for the file system, and quotas have been enabled with the **mmdefquotaon** command. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect. If **--perfileset-quota** is in effect, all users and groups in the fileset **root** will not be impacted by default quota unless they are explicitly set. The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options.

The **mmdefedquota** command displays the current values for these limits, if any, and prompts you to enter new values using your default editor:

- current block usage (display only)
- current inode usage (display only)
- inode soft limit
- inode hard limit
- block soft limit

Displayed in **KB**, but may be specified using **g, G, k, K, m, M, p, P, t, or T**. If no suffix is provided, the number is assumed to be in **bytes**.

- block hard limit

Displayed in **KB**, but may be specified using **g, G, k, K, m, M, p, P, t, or T**. If no suffix is provided, the number is assumed to be in **bytes**.

Note: A block or inode limit of 0 indicates no limit.

The **mmdefedquota** command waits for the edit window to be closed before checking and applying new values. If an incorrect entry is made, reissue the command and enter the correct values.

When you set quota limits for a file system, consider replication within the file system. For more information, see the topic *Listing quotas* in the *IBM Spectrum Scale: Advanced Administration Guide*.

The EDITOR environment variable must contain a complete path name, for example:

```
export EDITOR=/bin/vi
```


Parameters

Device

The device name of the file system to have default quota values set for.

File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Fileset

The name of a fileset in the file system to have default quota values set for.

Options

- g** Specifies that the default quota value is to be applied for new groups accessing the specified file system or fileset.
- j** Specifies that the default quota value is to be applied for new filesets in the specified file system.
- u** Specifies that the default quota value is to be applied for new users accessing the specified file system or fileset.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdefedquota** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmdefedquota** command is issued.

Examples

1. To set default quotas for new users of the file system **gpfs1**, issue this command:

```
mmdefedquota -u gpfs1
```

The system displays information in your default editor similar to:

```
*** Edit quota limits for USR DEFAULT entry
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs1: blocks in use: 0K, limits (soft = 0K, hard = 0K)
      inodes in use: 0, limits (soft = 0, hard = 0)
```

Change the soft block limit to 19 GB, the hard block limit to 20 GB, the inode soft limit to 1 KB, and the inode hard limit to 20 KB, as follows:

```
*** Edit quota limits for USR DEFAULT entry
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs1: blocks in use: 0K, limits (soft = 19G, hard = 20G)
      inodes in use: 0, limits (soft = 1K, hard = 20K)
```

After the edit window is closed, issue this command to confirm the change:

```
mmquota -d -u gpfs1
```

mmdefquota

The system displays information similar to:

Filesystem	type	Default Block Limits(KB)		Default File Limits	
		quota	limit	quota	limit
gpfs1	USR	19922944	20971520	1024	20480

- To set default quotas for new users of fileset **fset1** in file system **gpfs1**, issue this command:

```
mmdefquota -u gpfs1:fset1
```

The system displays information in your default editor similar to:

```
*** Edit quota limits for USR DEFAULT entry for fileset fset1
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs1: blocks in use: 0K, limits (soft = 0K, hard = 31457280K)
      inodes in use: 0, limits (soft = 0, hard = 0)
```

Change the soft block limit to 3 GB and the hard block limit to 6 GB, as follows:

```
*** Edit quota limits for USR DEFAULT entry for fileset fset1
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs1: blocks in use: 0K, limits (soft = 3G, hard = 6G)
      inodes in use: 0, limits (soft = 0, hard = 0)
```

After the edit window is closed, issue this command to confirm the change:

```
mmfsquota -d gpfs1:fset1
```

The system displays information similar to:

Filesystem	Fileset	type	Default Block Limits(KB)		Default File Limits		
			quota	limit	quota	limit	entryType
gpfs1	fset1	USR	3145728	6291456	0	0	default on
gpfs1	fset1	GRP	0	0	0	0	default on

See also

- “mmchfs command” on page 371
- “mmcheckquota command” on page 361
- “mmcrfs command” on page 414
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmfs command” on page 536
- “mmfsquota command” on page 559
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

```
/usr/lpp/mmfs/bin
```

mmdefquotaoff command

Deactivates default quota limit usage.

Synopsis

```
mmdefquotaoff [-u] [-g] [-j] [-v] [-d] {Device [Device...] | -a}
```

or

```
mmdefquotaoff [-u] [-g] [-v] [-d] {Device:Fileset ... | -a}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmdefquotaoff** command deactivates default quota limits for file systems and filesets. If default quota limits are deactivated, new users, groups, or filesets will then have a default quota limit of 0, indicating no limit.

If none of the following options are specified, the **mmdefquotaoff** command deactivates all default quotas:

- u
- j
- g

If the **-a** option is not used, *Device* must be the last parameter specified.

Parameters

Device

The device name of the file system to have default quota values deactivated.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Fileset

The name of a fileset in the file system to have default quota values deactivated.

Options

- a Deactivates default quotas for all GPFS file systems in the cluster. When used in combination with the **-g** option, only group quotas are deactivated. When used in combination with the **-u** or **-j** options, only user or fileset quotas, respectively, are deactivated.
- d Resets quota limits to zero for users, groups, or filesets.

When **--perfileset-quota** is not in effect for the file system, this option will reset quota limits to zero only for users, groups, or filesets that have default quotas established.

When **--perfileset-quota** is in effect for the file system, this option will reset quota limits to zero for users, groups, or filesets that have default quotas established only if *both* the file system and fileset-level default quotas are zero. If either file system or fileset-level default quotas exist, the default quotas will be switched to the level that is non-zero.

If this option is not chosen, existing quota entries remain in effect.
- g Specifies that default quotas for groups are to be deactivated.
- j Specifies that default quotas for filesets are to be deactivated.

mmdefquotaoff

- u Specifies that default quotas for users are to be deactivated.
- v Prints a message for each file system or fileset in which default quotas are deactivated.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdefquotaoff** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmdefquotaoff** command is issued.

Examples

1. To deactivate default user quotas on file system **fs0**, issue this command:

```
mmdefquotaoff -u fs0
```

To confirm the change, issue this command:

```
mmlsquota -d -u fs0
```

The system displays information similar to:

Filesystem type	Default Block Limits(KB)			Default File Limits		Remarks
	quota	limit		quota	limit	
fs0	USR	no default limits				

2. To deactivate default group quotas on all file systems, issue this command:

```
mmdefquotaoff -g -a
```

To confirm the change, issue this command:

```
mmlsquota -d -g
```

The system displays information similar to:

Filesystem type	Default Block Limits(KB)			Default File Limits		Remarks
	quota	limit		quota	limit	
fs0	GRP	no default limits				

Filesystem type	Default Block Limits(KB)			Default File Limits		Remarks
	quota	limit		quota	limit	
fs1	GRP	no default limits				

3. To deactivate both user and group default quotas for fileset **fset1** on file system **gpfs1**, issue this command:

```
mmdefquotaoff -d gpfs1:fset1
```

To confirm the change, issue this command:

```
mmlsquota -d gpfs1:fset1
```

The system displays information similar to:

Default Block Limits(KB)					Default File Limits		
Filesystem	Fileset	type	quota	limit	quota	limit	entryType
gpfs1	fset1	USR	0	0	0	0	default off
gpfs1	fset1	GRP	0	0	0	0	default off

See also

- “mmcheckquota command” on page 361
- “mmdefedquota command” on page 434
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmlsquota command” on page 559
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

/usr/lpp/mmfs/bin

mmdefquotaon command

Activates default quota limit usage.

Synopsis

```
mmdefquotaon [-u] [-g] [-j] [-v] [-d] {Device [Device... ] | -a}
```

or

```
mmdefquotaon [-u] [-g] [-v] [-d] {Device:Fileset ... | -a}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmdefquotaon** command activates default quota limits for file systems and filesets. If default quota limits are not applied, new users, groups, or filesets will have a quota limit of 0, indicating no limit.

To use default quotas, the **-Q yes** option must be in effect for the file system. To use default quotas at the fileset level, the **--perfileset-quota** option must also be in effect. The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options.

If none of the following options are specified, the **mmdefquotaon** command activates all default quota limits:

- u
- j
- g

If the **-a** option is not used, *Device* must be the last parameter specified.

Default quotas are established for new users, groups of users or filesets by issuing the **mmdefquota** command. Under the **-d** option, all users without an explicitly set quota limit will have a default quota limit assigned.

Parameters

Device

The device name of the file system to have default quota values activated.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Fileset

The name of a fileset in the file system to have default quota values activated.

Options

-a Activates default quotas for all GPFS file systems in the cluster. When used in combination with the **-g** option, only group quotas are activated. When used in combination with the **-u** or **-j** options, only user or fileset quotas, respectively, are activated.

-d Assigns default quota limits to existing users, groups, or filesets when the **mmdefquota** command is issued.

When **--perfileset-quota** is not in effect for the file system, this option will only affect existing users, groups, or filesets with no established quota limits.

When **--perfileset-quota** is in effect for the file system, this option will affect existing users, groups, or filesets with no established quota limits, and it will also change existing users or groups that refer to default quotas at the file system level into users or groups that refer to fileset-level default quota. For more information about default quota priorities, see the topic *Default quotas* in the *IBM Spectrum Scale: Advanced Administration Guide*.

If this option is not chosen, existing quota entries remain in effect and are not governed by the default quota rules.

- g Specifies that default quotas for groups are to be activated.
- j Specifies that default quotas for filesets are to be activated.
- u Specifies that default quotas for users are to be activated.
- v Prints a message for each file system or fileset in which default quotas are activated.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdfquotaon** command.

The node on which you enter the command must be able to execute remote shell commands on any other administration node in the cluster. It must be able to do so without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmdfquotaon** command is issued.

Examples

1. To activate default user quotas on file system **fs0**, issue this command:

```
mmdfquotaon -u fs0
```

To confirm the change, issue this command:

```
mmlsfs fs0 -Q
```

The system displays information similar to:

flag	value	description
-Q	user	Quotas enforced
	user	Default quotas enabled

2. To activate default group quotas on all file systems in the cluster, issue this command:

```
mmdfquotaon -g -a
```

To confirm the change, individually for each file system, issue this command:

```
mmlsfs fs1 -Q
```

The system displays information similar to:

flag	value	description
-Q	group	Quotas enforced
	group	Default quotas enabled

mmdefquotaon

- To activate user, group, and fileset default quotas on file system **fs2**, issue this command:

```
mmdefquotaon fs2
```

To confirm the change, issue this command:

```
mmlsfs fs2 -Q
```

The system displays information similar to:

flag	value	description
-Q	user;group;fileset	Quotas enforced
	user;group;fileset	Default quotas enabled

- To activate user default quota for fileset **fset1** on file system **gpfs1**, issue this command:

```
mmdefquotaon -d -u gpfs1:fset1
```

To confirm the change, issue this command:

```
mmlsquota -d gpfs1:fset1
```

The system displays information similar to:

Filesystem	Fileset	Default Block Limits(KB)		Default File Limits			
		type	quota	limit	quota	limit	entryType
gpfs1	fset1	USR	0	31457280	0	0	default on
gpfs1	fset1	GRP	0	0	0	0	default off

In this example, notice the entryType for user quota displays default on. To also activate group default quota for **fset1** on file system **gpfs1**, issue this command:

```
mmdefquotaon -d -g gpfs1:fset1
```

To confirm the change, issue this command:

```
mmlsquota -d gpfs1:fset1
```

The system displays information similar to :

Filesystem	Fileset	Default Block Limits(KB)		Default File Limits			
		type	quota	limit	quota	limit	entryType
gpfs1	fset1	USR	0	31457280	0	0	default on
gpfs1	fset1	GRP	0	0	0	0	default on

In this example, notice that the entryType for group quota also displays default on now.

See also

- “mmcheckquota command” on page 361
- “mmchfs command” on page 371
- “mmcrfs command” on page 414
- “mmdefedquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmedquota command” on page 481
- “mmlsfs command” on page 536
- “mmlsquota command” on page 559
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

```
/usr/lpp/mmfs/bin
```


mmdefragfs command

Reduces disk fragmentation by increasing the number of full free blocks available to the file system.

Synopsis

```
mmdefragfs Device [-i] [-u BlkUtilPct] [-P PoolName]
[-N {Node[,Node...] | NodeFile | NodeClass}] [--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdefragfs** command to reduce fragmentation of a file system. The **mmdefragfs** command moves existing file system data within a disk to make more efficient use of disk blocks. The data is migrated to unused sub-blocks in partially allocated blocks, thereby increasing the number of free full blocks.

The **mmdefragfs** command can be run against a mounted or unmounted file system. However, best results are achieved when the file system is unmounted. When a file system is mounted, allocation status may change causing retries to find a suitable unused sub-block.

Note: On a file system that has a very low level of fragmentation, negative numbers can be seen in the output of **mmdefragfs** for free sub-blocks. This indicates that the block usage has in fact increased after running the **mmdefragfs** command. If negative numbers are seen, it does not indicate a problem and you do not need to rerun the **mmdefragfs** command.

Parameters

Device

The device name of the file system to have fragmentation reduced. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-P *PoolName*

Specifies the pool name to use.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the nodes that can be used in this disk defragmentation. This parameter supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--qos *QOSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

mmdefragfs

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Options

-i Specifies to query the current disk fragmentation state of the file system. Does not perform the actual defragmentation of the disks in the file system.

-u *BlkUtilPct*

The average block utilization goal for the disks in the file system. The **mmdefragfs** command reduces the number of allocated blocks by increasing the percent utilization of the remaining blocks. The command automatically goes through multiple iterations until *BlkUtilPct* is achieved on all of the disks in the file system or until no progress is made in achieving *BlkUtilPct* from one iteration to the next, at which point it exits.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdefragfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To query the fragmentation state of file system **fs0**, issue this command:

```
mmdefragfs fs0 -i
```

The system displays information similar to:

disk name	disk size in nSubblk	in full blocks	subblk in fragments	% free blk	% blk util
nsd32	327680	277504	12931	84.688	96.054
nsd33	327680	315232	580	96.201	99.823
nsd21	327680	301824	2481	92.109	99.243
nsd34	327680	275904	13598	84.199	95.850
nsd30	327680	275808	13380	84.170	95.917
nsd19	327680	278496	12369	84.990	96.225
nsd31	327680	276224	12012	84.297	96.334
(total)	2293760	2000992	67351		97.064

2. To reduce fragmentation of the file system **fs0** on all defined, accessible disks that are not stopped or suspended, issue this command:

```
mmdefragfs fs0
```

The system displays information similar to:

disk name	free subblk in full blocks	free subblk in fragments	% free blk	% blk util		
	before	after freed	before	after	before	after
-----	-----	-----	-----	-----	-----	-----

gpfs57nsd	28896	29888	31	1462	463	50.39	52.12	94.86	98.31
gpfs60nsd	41728	43200	46	1834	362	59.49	61.59	93.55	98.66
(total)	70624	73088	77	3296	825			93.63	98.84

- To reduce fragmentation of all files in the **fs1** file system until the disks have 100% full block utilization, issue this command:

```
mmdefragfs fs1 -u 100
```

The system displays information similar to:

```
Defragmenting file system 'fs1'...
```

```
Defragmenting until full block utilization is 98.00%, currently 97.07%
27.35 % complete on Tue May 26 14:25:42 2009 ( 617882 inodes 4749 MB)
82.65 % complete on Tue May 26 14:26:02 2009 ( 1867101 inodes 10499 MB)
89.56 % complete on Tue May 26 14:26:23 2009 ( 2023206 inodes 14296 MB)
90.01 % complete on Tue May 26 14:26:43 2009 ( 2033337 inodes 17309 MB)
90.28 % complete on Tue May 26 14:27:03 2009 ( 2039551 inodes 19779 MB)
91.17 % complete on Tue May 26 14:27:23 2009 ( 2059629 inodes 23480 MB)
91.67 % complete on Tue May 26 14:27:43 2009 ( 2070865 inodes 26760 MB)
92.51 % complete on Tue May 26 14:28:03 2009 ( 2089804 inodes 29769 MB)
93.12 % complete on Tue May 26 14:28:23 2009 ( 2103697 inodes 32649 MB)
93.39 % complete on Tue May 26 14:28:43 2009 ( 2109629 inodes 34934 MB)
95.47 % complete on Tue May 26 14:29:04 2009 ( 2156805 inodes 36576 MB)
95.66 % complete on Tue May 26 14:29:24 2009 ( 2160915 inodes 38705 MB)
95.84 % complete on Tue May 26 14:29:44 2009 ( 2165146 inodes 40248 MB)
96.58 % complete on Tue May 26 14:30:04 2009 ( 2181719 inodes 41733 MB)
96.77 % complete on Tue May 26 14:30:24 2009 ( 2186053 inodes 43022 MB)
96.99 % complete on Tue May 26 14:30:44 2009 ( 2190955 inodes 43051 MB)
97.20 % complete on Tue May 26 14:31:04 2009 ( 2195726 inodes 43077 MB)
97.40 % complete on Tue May 26 14:31:24 2009 ( 2200378 inodes 43109 MB)
97.62 % complete on Tue May 26 14:31:44 2009 ( 2205201 inodes 43295 MB)
97.83 % complete on Tue May 26 14:32:05 2009 ( 2210003 inodes 43329 MB)
97.85 % complete on Tue May 26 14:32:25 2009 ( 2214741 inodes 43528 MB)
97.86 % complete on Tue May 26 14:32:55 2009 ( 2221888 inodes 43798 MB)
97.87 % complete on Tue May 26 14:33:35 2009 ( 2231453 inodes 44264 MB)
97.88 % complete on Tue May 26 14:34:26 2009 ( 2243181 inodes 45288 MB)
100.00 % complete on Tue May 26 14:35:10 2009
```

disk name	free subblk in full blocks		blk freed	free subblk in fragments		% free blk		% blk util	
	before	after		before	after	before	after	before	after
nsd32	277504	287840	323	12931	2183	84.69	87.84	96.05	99.33
nsd33	315232	315456	7	580	185	96.20	96.27	99.82	99.94
nsd21	301824	303616	56	2481	666	92.11	92.66	99.24	99.80
nsd34	275904	285920	313	13598	3159	84.20	87.26	95.85	99.04
nsd30	275840	285856	313	13348	2923	84.18	87.24	95.93	99.11
nsd19	278592	288832	320	12273	1874	85.02	88.14	96.25	99.43
nsd31	276224	284608	262	12012	3146	84.30	86.86	96.33	99.04
(total)	2001120	2052128	1594	67223	14136			97.07	99.38

Defragmentation complete, full block utilization is 99.04%.

See also

- “mmdf command” on page 470

Location

```
/usr/lpp/mmfs/bin
```

mmdelacl

mmdelacl command

Deletes a GPFS access control list.

Synopsis

```
mmdelacl [-d] Filename
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmdelacl** command to delete the extended entries of an access ACL of a file or directory, or to delete the default ACL of a directory.

Parameters

Filename

The path name of the file or directory for which the ACL is to be deleted. If the **-d** option is specified, *Filename* must contain the name of a directory.

Options

-d Specifies that the default ACL of a directory is to be deleted.

Since there can be only one NFS V4 ACL (no separate default), specifying the **-d** flag for a file with an NFS V4 ACL is an error. Deleting an NFS V4 ACL necessarily removes both the ACL and any inheritable entries contained in it.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

The **mmdelacl** command may be issued only by the file or directory owner, the root user, or by someone with control (c) authority in the ACL for the file.

You may issue the **mmdelacl** command only from a node in the GPFS cluster where the file system is mounted.

Examples

To delete the default ACL for a directory named **project2**, issue this command:

```
mmdelacl -d project2
```

To confirm the deletion, issue this command:

```
mmgetacl -d project2
```

The system displays information similar to:

```
#owner:uno  
#group:system
```

See also

- “mmeditACL command” on page 478
- “mmgetACL command” on page 500
- “mmputACL command” on page 614

Location

/usr/lpp/mmfs/bin

mmdelcallback command

Deletes one or more user-defined callbacks from the GPFS system.

Synopsis

```
mmdelcallback CallbackIdentifier[,CallbackIdentifier...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdelcallback** command to delete one or more user-defined callbacks from the GPFS system.

Parameters

CallbackIdentifier

Specifies a user-defined unique name that identifies the callback to be deleted. Use the **mmlscallback** command to see the name of the callbacks that can be deleted.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdelcallback** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To delete the **test1** callback from the GPFS system, issue this command:

```
mmdelcallback test1
```

The system displays information similar to:

```
mmdelcallback: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

See also

- “mmaddcallback command” on page 226
- “mmlscallback command” on page 522

Location

```
/usr/lpp/mmfs/bin
```

mmdeldisk command

Deletes disks from a GPFS file system.

Synopsis

```
mmdeldisk Device {"DiskName[;DiskName...]" | -F DescFile} [-a] [-c]
[-m | -r | -b] [-N {Node[,Node...]} | NodeFile | NodeClass}]
[--inode-criteria CriteriaFile] [-o InodeResultFile]
[--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmdeldisk** command migrates all data that would otherwise be lost to the remaining disks in the file system. It then removes the disks from the file system descriptor, preserves replication at all times, and optionally rebalances the file system after removing the disks.

The **mmdeldisk** command has the following two functions:

- Copying unreplicated data off the disks and removing references to the disks (**deldisk** step).
- Rereplicating or rebalancing blocks across the remaining disks (**restripe** step).

These two functions can be done in one pass over the file system, or in two passes if the **-a** option is specified.

Run the **mmdeldisk** command when system demand is low.

If a replacement for a failing disk is available, use the **mmrpldisk** command in order to keep the file system balanced. Otherwise, use one of these procedures to delete a disk:

- If the file system is replicated, replica copies can be preserved at all times by using the default **-r** option or the **-b** option.
- Using the **-m** option will not preserve replication during the **deldisk** step because it will only copy the minimal amount of data off the disk being deleted so that every block has at least one copy. Also, using the **-a** option will not preserve replication during the **deldisk** step, but will then re-establish replication during the subsequent **restripe** step.
- If you want to move all data off the disk before running **mmdeldisk**, use **mmchdisk** to suspend all the disks that will be deleted and run **mmrestripefs** with the **-r** or **-b** option. This step is no longer necessary, now that **mmdeldisk** does the same function. If **mmdeldisk** fails (or is canceled), it leaves the disks in the suspended state, and **mmdeldisk** can be retried when the problem that caused **mmdeldisk** to stop is corrected.
- If the disk is permanently damaged and the file system is not replicated, or if the **mmdeldisk** command repeatedly fails, see the *IBM Spectrum Scale: Problem Determination Guide* and search for *Disk media failure*.

If the last disk in a storage pool is deleted, the storage pool is deleted. The **mmdeldisk** command is not permitted to delete the **system** storage pool. A storage pool must be empty in order for it to be deleted.

Results

Upon successful completion of the **mmdeldisk** command, these tasks are completed:

- Data that has not been replicated from the target disks is migrated to other disks in the file system.
- Remaining disks are rebalanced, if specified.

mmdeldisk

Parameters

Device

The device name of the file system to delete the disks from. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**. This must be the first parameter.

"DiskName[;DiskName...]"

Specifies the names of the disks to be deleted from the file system. If there is more than one disk to be deleted, delimit each name with a semicolon (;) and enclose the list in quotation marks.

-F DiskFile

Specifies a file that contains the names of the disks (one name per line), to be deleted from the GPFS cluster.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes that participate in the restripe of the file system after the specified disks have been removed. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--inode-criteria CriteriaFile

Specifies the interesting inode criteria flag, where *CriteriaFile* is one of the following:

BROKEN

Indicates that a file has a data block with all of its replicas on disks that have been removed.

Note: **BROKEN** is always included in the list of flags even if it is not specified.

dataUpdateMiss

Indicates that at least one data block was not updated successfully on all replicas.

exposed

Indicates an inode with an exposed risk; that is, the file has data where all replicas are on suspended disks. This could cause data to be lost if the suspended disks have failed or been removed.

i11Compressed

Indicates an inode in which file compression or decompression is deferred, or in which a compressed file is partly decompressed to allow the file to be written into or memory-mapped.

i11Placed

Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

i11Replicated

Indicates that the file has a data block that does not meet the setting for the replica.

metaUpdateMiss

Indicates that there is at least one metadata block that has not been successfully updated to all replicas.

unbalanced

Indicates that the file has a data block that is not well balanced across all the disks in all failure groups.

Note: If a file matches *any* of the specified interesting flags, all of its interesting flags (even those not specified) will be displayed.

-o InodeResultFile

Contains a list of the inodes that met the interesting inode flags that were specified on the **--inode-criteria** parameter. The output file contains the following:

INODE_NUMBER

This is the inode number.

DISKADDR

Specifies a dummy address for later **tsfindinode** use.

SNAPSHOT_ID

This is the snapshot ID.

ISGLOBAL_SNAPSHOT

Indicates whether or not the inode is in a global snapshot. Files in the live file system are considered to be in a global snapshot.

INDEPENDENT_FSETID

Indicates the independent fileset to which the inode belongs.

MEMO (INODE_FLAGS FILE_TYPE [ERROR])

Indicates the inode flag and file type that will be printed:

Inode flags:

BROKEN
exposed
dataUpdateMiss
i11Compressed
i11Placed
i11Replicated
metaUpdateMiss
unbalanced

File types:

BLK_DEV
CHAR_DEV
DIRECTORY
FIFO
LINK
LOGFILE
REGULAR_FILE
RESERVED
SOCK
UNLINKED
DELETED

Notes:

1. An error message will be printed in the output file if an error is encountered when repairing the inode.
2. **DISKADDR**, **ISGLOBAL_SNAPSHOT**, and **FSET_ID** work with the **tsfindinode** tool (`/usr/lpp/mmfs/bin/tsfindinode`) to find the file name for each inode. **tsfindinode** uses the output file to retrieve the file name for each interesting inode.

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other

This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

mmdeldisk

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Options

- a Specifies that the **mmdeldisk** command *not* wait for rereplicating or rebalancing to complete before returning. When this flag is specified, the **mmdeldisk** command runs asynchronously and returns after the file system descriptor is updated and the rebalancing scan is started, but it does not wait for rebalancing to finish. If no rebalancing is requested (**-r** option is not specified), this option has no effect.

If **-m** is specified, this option has no effect. If **-r** or **-b** is specified (no option defaulting to **-r**), then the **deldisk** step is done using **-m**, and the **restripe** step is done using the specified option.

- b Rebalances the blocks onto the other disks while moving data off the disks being deleted. This might have to move much more data than the **-r** operation.

Note: Rebalancing of files is an I/O intensive and time consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance your file system over time, without the cost of the rebalancing.

- c Specifies that processing continues even in the event that unreadable data exists on the disks being deleted. Data that has not been replicated is lost. Replicated data is not lost as long as the disks containing the replication are accessible.
- m Does minimal data copying to preserve any data that is located only on the disks being removed. This is the fastest way to get a disk out of the system, but it could reduce replication of some blocks of the files and metadata.

Note: This might be I/O intensive if there is a lot of data to be copied or rereplicated off the disks that are being deleted.

- r Preserves replication of all files and metadata during the **mmdeldisk** operation (except when the **-a** option is specified). This is the default.

Note: This might be I/O intensive if there is a lot of data to be copied or rereplicated off the disks that are being deleted.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdeldisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To delete **gpfs1016nsd** from file system **fs1** and rebalance the files across the remaining disks, issue this command:

```
mmdeldisk fs1 gpfs1016nsd
```

The system displays information similar to:

```

Deleting disks ...
Scanning spl storage pool
Scanning user file metadata ...
 100.00 % complete on Tue Mar 13 15:48:51 2012
Scan completed successfully.
Checking Allocation Map for storage pool 'spl'
tsdeldisk64 completed.
mmdeldisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```

2. To forcibly delete disk **vmip2_nsd3**, from file system **fs1**, issue this command:

```
mmdeldisk fs1 vmip2_nsd3 -p
```

The system displays information similar to:

```

Deleting disks ...
Checking Allocation Map for storage pool system
 68 % complete on Wed Apr 15 10:09:30 2015
 100 % complete on Wed Apr 15 10:09:32 2015
Checking Allocation Map for storage pool data
 73 % complete on Wed Apr 15 10:09:37 2015
 100 % complete on Wed Apr 15 10:09:39 2015
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
Scanning file system metadata for data storage pool
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...
 100.00 % complete on Wed Apr 15 10:09:48 2015 \
(65792 inodes with total      397 MB data processed)
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-3908 Check file '/var/mmfs/tmp/fs1.pit.interestingInodes.12884901889' \
on vmip1 for inodes with broken disk addresses or failures.
GPFS: 6027-000 Attention: A disk being removed reduces the number of failure groups to 1,
which is below the number required for replication: 2.
New blocks will be allocated from the remaining disks,
but files will be unreplicated and hence at risk.
GPFS: 6027-370 tsdeldisk completed.
mmdeldisk: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
#10:10:43# vmip1:~ # cat /var/mmfs/tmp/fs1.pit.interestingInodes.12884901889
This inode list was generated in the Parallel Inode Traverse on Wed Apr 15 10:09:47 2015
INODE_NUMBER DISKADDR SNAPSHOT_ID ISGLOBAL_SNAPSHOT FSET_ID MEMO(INODE_FLAGS FILE_TYPE [ERROR])
57088      0:0      0      1      0      broken REGULAR_FILE Error: 226, Input/output error
57091      0:0      0      1      0      broken REGULAR_FILE Error: 226, Input/output error
57093      0:0      0      1      0      broken REGULAR_FILE Error: 226, Input/output error
57096      0:0      0      1      0      broken REGULAR_FILE Error: 226, Input/output error

```

3. To identify the file names matching the "interesting" criteria (in this case, having broken disk addresses), issue this command:

```
tsfindinode -i /var/mmfs/tmp/fs1.pit.interestingInodes.12884901889 /fs1
```

The system displays information similar to:

This inode list was generated in the Parallel Inode Traverse on Wed Apr 15 10:11:47 2015

```

INODE_NUMBER DISKADDR SNAPSHOT_ID ISGLOBAL_SNAPSHOT FSET_ID MEMO(INODE_FLAGS FILE_TYPE [ERROR])

57096      0      0xD4EB /fs1/file9
57088      0      0xC0CC /fs1/file1
57091      0      0x6788 /fs1/file4
57093      0      0x93F4 /fs1/file6

```

mmdeldisk

See also

- “mmaddisk command” on page 239
- “mmchdisk command” on page 354
- “mmlsdisk command” on page 528
- “mmrpldisk command” on page 648

Location

/usr/lpp/mmfs/bin

mmdelfileset command

Deletes a GPFS fileset.

Synopsis

```
mmdelfileset Device FilesetName [-f] [--qos QoSClass]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmdelfileset** command deletes a GPFS fileset. When deleting a fileset, consider these points:

- The root fileset cannot be deleted.
- A fileset that is not empty cannot be deleted unless the **-f** flag is specified.
- A fileset that is currently linked into the namespace cannot be deleted until it is unlinked with the **mmunlinkfileset** command.
- A dependent fileset can be deleted at any time.
- An independent fileset cannot be deleted if it has any dependent filesets or fileset snapshots.
- Deleting a dependent fileset that is included in a fileset or global snapshot removes it from the active file system, but it remains part of the file system in a deleted state.
- Deleting an independent fileset that is included in any global snapshots removes it from the active file system, but it remains part of the file system in a deleted state.
- A fileset in the deleted state is displayed in the **mmlsfileset** output with the fileset name in parenthesis. If the **-L** flag is specified, the latest including snapshot is also displayed. The **--deleted** option of the **mmlsfileset** command can be used to display only deleted filesets.
- The contents of a deleted fileset are still available in the snapshot, through some path name containing a **.snapshots** component, because it was saved when the snapshot was created.
- When the last snapshot that includes the fileset has been deleted, the fileset is fully removed from the file system.

For information on GPFS filesets, see *Information Lifecycle Management for GPFS in IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system that contains the fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

FilesetName

Specifies the name of the fileset to be deleted.

-f Forces the deletion of the fileset. All fileset contents are deleted. Any child filesets are first unlinked.

--qos *QoSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

mmdelfileset

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdelfileset** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This sequence of commands illustrates what happens when attempting to delete a fileset that is linked.

a. Command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status      Path
root           Linked     /gpfs1
fset1          Linked     /gpfs1/fset1
fset2          Unlinked --
```

b. Command:

```
mmdelfileset gpfs1 fset1
```

The system displays output similar to:

```
Fileset fset1 must be unlinked to be deleted.
```

c. Command:

```
mmdelfileset gpfs1 fset2
```

The system displays output similar to:

```
Checking fileset ...
Checking fileset complete.
Deleting fileset ...
Fileset 'fset2' deleted.
```

d. To confirm the change, issue this command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status      Path
root           Linked     /gpfs1
fset1          Linked     /gpfs1/fset1
```

2. This sequence of commands illustrates what happens when attempting to delete a fileset that contains user files.

- a. Command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status   Path
root           Linked  /gpfs1
fset1          Linked  /gpfs1/fset1
fset2          Unlinked --
```

- b. Command:

```
mmdelfileset gpfs1 fset2
```

The system displays output similar to:

```
Fileset 'fset2' contains user files,
but can be deleted with the "-f" option.
```

- c. Command:

```
mmdelfileset gpfs1 fset2 -f
```

The system displays output similar to:

```
Checking fileset ...
Checking fileset complete.
Deleting user files ...
100.00 % complete on Wed Feb 15 11:38:05 2012
Deleting fileset ...
Fileset 'fset2' deleted.
```

- d. To confirm the change, issue this command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status   Path
root           Linked  /gpfs1
fset1          Linked  /gpfs1/fset1
```

See also

- “mmchfileset command” on page 365
- “mmcrfileset command” on page 408
- “mmlinkfileset command” on page 517
- “mmlsfileset command” on page 532
- “mmunlinkfileset command” on page 687

Location

```
/usr/lpp/mmfs/bin
```

mmdelfs command

Removes a GPFS file system.

Synopsis

```
mmdelfs Device [-p]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmdelfs** command removes all the structures for the specified file system from the nodes in the cluster.

Before you can delete a file system using the **mmdelfs** command, you must unmount it on all nodes.

Results

Upon successful completion of the **mmdelfs** command, these tasks are completed on all nodes:

- Deletes the character device entry from **/dev**.
- Removes the mount point directory where the file system had been mounted.

Parameters

Device

The device name of the file system to be removed. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-p Indicates that the disks are permanently damaged and the file system information should be removed from the GPFS cluster data even if the disks cannot be marked as **available**.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdelfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To delete file system **fs0**, issue this command:

```
mmdelfs fs0
```

The system displays information similar to:


```
mmdelfs: 6027-1366 Marking the disks as available
GPFS: 6027-573 All data on the following disks of fs0 will be destroyed:
  gpfs9nsd
  gpfs10nsd
  gpfs15nsd
  gpfs17nsd
GPFS: 6027-574 Completed deletion of file system fs0.
mmdelfs: 6027-1371 Propagating the cluster configuration data to all
  affected nodes. This is an asynchronous process.
```

See also

- “mmcrfs command” on page 414
- “mmchfs command” on page 371
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmdelnode command

Removes one or more nodes from a GPFS cluster.

Synopsis

```
mmdelnode {-a | -N Node[,Node...] | NodeFile | NodeClass}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdelnode** command to delete one or more nodes from the GPFS cluster. You may issue the **mmdelnode** command on any GPFS node.

A node cannot be deleted if any of the following are true:

1. It is a primary or secondary GPFS cluster configuration server.
The node being deleted cannot be the primary or secondary GPFS cluster configuration server unless you intend to delete the entire cluster.
You can determine whether a node is the primary or secondary configuration server by issuing the **mmlscluster** command. If the node is listed as of the servers and you still want to delete it without deleting the cluster, first use the **mmchcluster** command to assign another node as the server.
2. It is defined as an NSD server.
The node being deleted cannot be defined as an NSD server for any disk unless you intend to delete the entire cluster.
You can determine whether a node is an NSD server for one or more disks by issuing the **mmlsnsd** command. If the node is listed as an NSD server and you still want to delete it without deleting the cluster, first use the **mmchnsd** command to assign another node as an NSD server for the affected disks.
3. If the GPFS state is *unknown* and the node is reachable on the network.
You cannot delete a node if both of the following are true:
 - The node responds to a TCP/IP ping command from another node.
 - The status of the node shows *unknown* when you use the **mmgetstate** command from another node in the cluster.

Note: You will probably be able to delete such a node if you physically power it off.

You must follow these rules when deleting nodes:

1. Unless all nodes in the cluster are being deleted, run the **mmdelnode** command from a node that will remain in the cluster.
2. Before you can delete a node, unmount all of the GPFS file systems and stop GPFS on the node to be deleted.
3. Exercise caution when shutting down GPFS on quorum nodes. If the number of remaining quorum nodes falls below the requirement for a quorum, you will be unable to perform file system operations. For more information, see “quorum” in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Each GPFS cluster is managed independently, so there is no automatic coordination and propagation of changes between clusters like there is between the nodes within a cluster. This means that if you permanently delete nodes that are being used as contact nodes by other GPFS clusters that can mount your file systems, you should notify the administrators of those GPFS clusters so that they can update their own environments.

Results

Upon successful completion of the **mmdelnode** command, the specified nodes are deleted from the GPFS cluster.

Parameters

-a Delete all nodes in the cluster.

-N *{Node[,Node...] | NodeFile | NodeClass}*
Specifies the set of nodes to be deleted from the cluster.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

Exit status

0 Successful completion.

nonzero
A failure has occurred.

Security

You must have root authority to run the **mmdelnode** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

You may issue the **mmdelnode** command from any node that will remain in the GPFS cluster.

Examples

- To delete all of the nodes in the cluster, issue this command:

```
mmdelnode -a
```

The system displays information similar to:

```
Verifying GPFS is stopped on all affected nodes ...
mmdelnode: Command successfully completed
mmdelnode: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

- To delete nodes **k145n12**, **k145n13**, and **k145n14**, issue this command:

```
mmdelnode -N k145n12,k145n13,k145n14
```

The system displays information similar to:

```
Verifying GPFS is stopped on all affected nodes ...
mmdelnode: Command successfully completed
mmdelnode: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

See also

- “mmaddnode command” on page 245
- “mmcrcluster command” on page 403
- “mmchconfig command” on page 331
- “mmlsfs command” on page 536

mmdelnode

- “mmlscluster command” on page 524

Location

/usr/lpp/mmfs/bin

mmdelnodeclass command

Deletes user-defined node classes.

Synopsis

```
mmdelnodeclass ClassName [,ClassName...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdelnodeclass** command to delete existing user-defined node classes.

Parameters

ClassName

Specifies an existing user-defined node class to delete.

If *ClassName* was used to change configuration attributes with **mmchconfig**, and the configuration attributes are still referencing *ClassName*, then *ClassName* cannot be deleted. Use the **mmchconfig** command to remove the references to *ClassName* before deleting this user-defined node class.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdelnodeclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To display the current user-defined node classes, issue this command:

```
mmfnodeclass --user
```

The system displays information similar to:

Node Class Name	Members
siteA	c6f1c3vp1,c6f1c3vp2
siteB	c6f1c3vp4,c6f1c3vp5

To delete the **siteA** node class, issue this command:

```
mmdelnodeclass siteA
```

The system displays information similar to:

```
mmdelnodeclass: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

mmdelnodeclass

To display the updated list of user-defined node classes, issue this command:

```
mmlsnodeclass --user
```

The system displays information similar to:

Node Class Name	Members
siteB	c6f1c3vp4,c6f1c3vp5

See also

- “mmcrnodeclass command” on page 424
- “mmchnodeclass command” on page 385
- “mmlsnodeclass command” on page 546

Location

```
/usr/lpp/mmfs/bin
```

mmdelnsd command

Deletes Network Shared Disks (NSDs) from the GPFS cluster.

Synopsis

```
mmdelnsd {"DiskName[;DiskName...]" | -F DiskFile}
```

or

```
mmdelnsd -p NSDId [-N Node[,Node...]]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmdelnsd** command serves two purposes:

1. To delete NSDs from the GPFS cluster.
2. To remove the unique NSD volume ID left on a disk after the failure of a previous invocation of the **mmdelnsd** command. The NSD had been successfully deleted from the GPFS cluster but there was a failure to clear sector 2 of the disk.

NSDs being deleted cannot be part of any file system. To determine if an NSD belongs to a file system or not, issue the **mmisnsd -d DiskName** command. If an NSD belongs to a file system, either the **mmdeldisk** or the **mmdelfs** command must be issued prior to deleting the NSDs from the GPFS cluster.

NSDs being deleted cannot be tiebreaker disks. To list the tiebreaker disks, issue the **mmisconfig tiebreakerDisks** command. Use the **mmchconfig** command to assign new tiebreaker disks prior to deleting NSDs from the cluster. For information on tiebreaker disks, see the "Quorum" topic in the *IBM Spectrum Scale Concepts, Planning, and Installation Guide*.

Results

Upon successful completion of the **mmdelnsd** command, these tasks are completed:

- All references to the disks are removed from the GPFS cluster data.
- Sector 2 of each disk is cleared of its unique NSD volume ID.
- On Windows, the disk's GPT partition table is removed leaving the disk Unknown/Not Initialized.

Parameters

DiskName[;DiskName...]

Specifies the names of the NSDs to be deleted from the GPFS cluster. Specify the names generated when the NSDs were created. Use the **mmisnsd -F** command to display disk names. If there is more than one disk to be deleted, delimit each name with a semicolon (;) and enclose the list of disk names in quotation marks.

-F *DiskFile*

Specifies a file containing the names of the NSDs, one per line, to be deleted from the GPFS cluster.

-N *Node[,Node...]*

Specifies the nodes to which the disk is attached. If no nodes are listed, the disk is assumed to be directly attached to the local node.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

mmdelnsd

-p *NSDI*

Specifies the NSD volume ID of an NSD that needs to be cleared from the disk as indicated by the failure of a previous invocation of the **mmdelnsd** command.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdelnsd** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To delete **gpfs47nsd** from the GPFS cluster, issue this command:

```
mmdelnsd "gpfs47nsd"
```

The system displays output similar to:

```
mmdelnsd: Processing disk gpfs47nsd
mmdelnsd: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

2. If after running **mmdelnsd** to delete an NSD, you experience a failure, the disk was not found. Run **mmdelnsd -p** *NSD Volume ID*. For example:

```
| mmdelnsd -p C0A8910B626630E
```

```
| This will remove the NSD definition from the GPFS configuration even if the NSD ID is not removed
| from the physical disk because it has been permanently lost.
```

See also

- “mmcrnsd command” on page 426
- “mmlsnsd command” on page 549

Location

```
/usr/lpp/mmfs/bin
```


mmdelsnapshot command

Deletes a GPFS snapshot.

Synopsis

```
mmdelsnapshot Device SnapshotName [-j FilesetName]
    [-N {Node[,Node...] | NodeFile | NodeClass}]
    [--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdelsnapshot** command to delete a GPFS snapshot.

Once the command is issued, the snapshot is marked for deletion and cannot be recovered.

If the node from which the **mmdelsnapshot** command is issued or the file system manager node fails, the snapshot might not be completely deleted. The **mmlssnapshot** command shows these snapshots with status `DeleteRequired`. Reissue the **mmdelsnapshot** from another node to complete the removal, or allow the snapshot to be cleaned up automatically by a later **mmdelsnapshot** command. A snapshot in this state cannot be accessed.

Any files open in the snapshot are forcibly closed. The user receives an **errno** of **ESTALE** on the next file access.

If a snapshot has file clones, you must delete the file clones or split them from their clone parents before you delete the snapshot. Use the **mmclone split** or **mmclone redirect** command to split file clones. Use a regular delete (**rm**) command to delete a file clone. If a snapshot is deleted that contains a clone parent, any attempts to read a block that refers to the missing snapshot returns an error. A policy file can be created to help determine whether a snapshot has file clones. See the *IBM Spectrum Scale: Advanced Administration Guide* for more information about file clones and policy files.

Parameters

Device

The device name of the file system for which the snapshot is to be deleted. File system names do not need to be fully qualified.

SnapshotName

Specifies the name of the snapshot to be deleted.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the nodes that participate in deleting the snapshot. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-j *FilesetName*

Specifies the name of the fileset that contains the fileset snapshot to be deleted (*SnapshotName*). If **-j** is not specified, the **mmdelsnapshot** command attempts to delete a global snapshot named *SnapshotName*.

--qos *QOSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command

mmdelsnapshot

is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

Nonzero

A failure occurred.

Security

You must have root authority to run the **mmdelsnapshot** command when you delete global snapshots.

Independent fileset owners can run the **mmdelsnapshot** command to delete snapshots of filesets that they own.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To delete the snapshot **snap1** for the file system **fs1**, issue the following command:

```
mmdelsnapshot fs1 snap1
```

The output is similar to the following example:

```
Invalidating snapshot files...
Deleting snapshot files...
 100.00 % complete on Tue Feb 28 10:40:59 2012
Delete snapshot snap1 complete, err = 0
```

Before you issue the command, the directory might have the following structure:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3

/fs1/.snapshots/snap1/file1
/fs1/.snapshots/snap1/userA/file2
/fs1/.snapshots/snap1/userA/file3
```

After you issue the command, the directory has the following structure:

```
/fs1/file1  
/fs1/userA/file2  
/fs1/userA/file3  
/fs1/.snapshots
```

See also

- “mmclone command” on page 400
- “mmcrsnapshot command” on page 431
- “mmlssnapshot command” on page 563
- “mmrestorefs command” on page 635
- “mmsnapdir command” on page 674

Location

```
/usr/lpp/mmfs/bin
```

mmdf command

Queries available file space on a GPFS file system.

Synopsis

```
mmdf Device [-d] [-F] [-m] [-P PoolName] [--block-size {BlockSize | auto}]
      [--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdf** command to display available file space on a GPFS file system. For each disk in the GPFS file system, the **mmdf** command displays this information, by failure group and storage pool:

- The size of the disk.
- The failure group of the disk.
- Whether the disk is used to hold data, metadata, or both.
- Available space in full blocks.
- Available space in fragments.

Displayed values are rounded down to a multiple of 1024 bytes. If the fragment size used by the file system is not a multiple of 1024 bytes, then the displayed values may be lower than the actual values. This can result in the display of a total value that exceeds the sum of the rounded values displayed for individual disks. The individual values are accurate if the fragment size is a multiple of 1024 bytes.

For the file system, the **mmdf** command displays the total number of inodes and the number available.

The **mmdf** command may be run against a mounted or unmounted file system.

Notes:

1. This command is I/O intensive and should be run when the system load is light.
2. An asterisk at the end of a line means that this disk is in a state where it is not available for new block allocation.

Parameters

Device

The device name of the file system to be queried for available file space. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-d List only disks that can hold data.

-F List the number of inodes and how many of them are free.

-m List only disks that can hold metadata.

-P *PoolName*

Lists only disks that belong to the requested storage pool.

--block-size {*BlockSize* | **auto**}

Specifies the unit in which the number of blocks is displayed. The value must be of the form [*n*]**K**, [*n*]**M**, [*n*]**G** or [*n*]**T**, where *n* is an optional integer in the range 1 to 1023. The default is 1K. If **auto** is specified, the number of blocks is automatically scaled to an easy-to-read value.

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

If you are a non-root user, you may specify only file systems that belong to the same cluster as the node on which the **mmdf** command was issued.

Examples

- To query all disks in the **fs2** file system that can hold data, issue this command:

```
mmdf fs2 -d
```

The system displays information similar to:

disk name	disk size in KB	failure group	holds metadata	holds data	free KB full blocks	free KB in fragments

Disks in storage pool: system (Maximum disk size allowed is 833 GB)						

(pool total)	0				0 (0%)	0 (0%)
Disks in storage pool: sp1 (Maximum disk size allowed is 359 GB)						
gpfs1002nsd	8897968	1	no	yes	8342016 (94%)	928 (0%)

(pool total)	8897968				8342016 (94%)	928 (0%)
(data)	8897968				8342016 (94%)	928 (0%)
(metadata)	0				0 (0%)	0 (0%)

(total)	8897968				8342016 (94%)	928 (0%)

- To query all disks in the **fs1** file system with the number of blocks automatically scaled to an easy-to-read value, issue this command:

```
mmdf fs1 --block-size auto
```

mmdf

The system displays information similar to:

disk name	disk size	failure group	holds metadata	holds data	free in full blocks	free in fragments
Disks in storage pool: system (Maximum disk size allowed is 437 GB)						
gpfs1001nsd	33.4G	1	yes	no	33.19G (99%)	2.5M (0%)
gpfs1002nsd	33.4G	1	yes	no	33.2G (99%)	2.5M (0%)
gpfs1003nsd	33.4G	2	yes	no	33.2G (99%)	2.5M (0%)
gpfs1004nsd	33.4G	2	yes	no	33.27G (100%)	2.5M (0%)
gpfs1005nsd	33.4G	2	yes	no	33.32G (100%)	1.562M (0%)
(pool total)	167G				166.2G (100%)	11.56M (0%)
Disks in storage pool: sp1 (Maximum disk size allowed is 484 GB)						
gpfs1006nsd	33.4G	4	no	yes	33.4G (100%)	1.562M (0%)
gpfs1007nsd	33.4G	4	no	yes	33.4G (100%)	1.562M (0%)
gpfs1008nsd	33.4G	4	no	yes	33.4G (100%)	1.562M (0%)
gpfs1010nsd	33.4G	4	no	yes	33.4G (100%)	1.562M (0%)
gpfs1011nsd	33.4G	4	no	yes	33.4G (100%)	1.562M (0%)
gpfs1012nsd	33.4G	5	no	yes	33.4G (100%)	1.562M (0%)
gpfs1013nsd	33.4G	5	no	yes	33.4G (100%)	1.562M (0%)
gpfs1014nsd	33.4G	5	no	yes	33.4G (100%)	1.562M (0%)
gpfs1015nsd	33.4G	5	no	yes	33.4G (100%)	1.562M (0%)
gpfs1016nsd	33.4G	5	no	yes	33.4G (100%)	1.562M (0%)
(pool total)	334G				334G (100%)	15.62M (0%)
(data)	334G				334G (100%)	15.62M (0%)
(metadata)	167G				166.2G (100%)	11.56M (0%)
(total)	501G				500.2G (100%)	27.19M (0%)

Inode Information

```
-----
Number of used inodes:      4043
Number of free inodes:     497717
Number of allocated inodes: 501760
Maximum number of inodes:  514048
```

- To query **fs1** for inode information, issue this command:

```
mmdf fs1 -F
```

The system displays information similar to:

Inode Information

```
-----
Number of used inodes:      4043
Number of free inodes:     497717
Number of allocated inodes: 501760
Maximum number of inodes:  514048
```

See also

- “mmchfs command” on page 371
- “mmcrfs command” on page 414
- “mmdelfs command” on page 458
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmdiag command

Displays diagnostic information about the internal GPFS state on the current node.

Synopsis

```
mmdiag [--all] [--version] [--waiters] [--deadlock] [--threads]
        [--memory] [--network] [--config] [--trace] [--assert]
        [--iohist] [--tokenmgr] [--commands] [--lroc]
        [--dmapi [session|event|token|disposition|all]]
        [--rpc [node[=name] | size|message|all|nn{S|s|M|m|H|h|D|d}]]
        [--stats]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmdiag** command to query various aspects of the GPFS internal state for troubleshooting and tuning purposes. The **mmdiag** command displays information about the state of GPFS on the node where it is executed. The command obtains the required information by querying the GPFS daemon process (**mmfsd**), and thus will only function when the GPFS daemon is running.

Results

The **mmdiag** command displays the requested information and returns 0 if successful.

Parameters

--all

Displays all available information. This is the same as specifying all of the **mmdiag** parameters.

--assert

Display current dynamic assert status and levels.

--commands

Displays all the commands currently running on the local node.

--config

Displays configuration parameters and their settings. The list of configuration parameters shown here consists of configuration parameters known to **mmfsd**. Note that some configuration parameters (for example, trace settings) are only perused by the layers of code above **mmfsd**, and those will be shown in **mmlsconfig** output, but not here.

On the other hand, while **mmlsconfig** only displays a subset of configuration parameters (generally those that have nondefault settings), the list here shows a larger parameter set. All of the documented **mmfsd** configuration parameters are shown, plus some of the undocumented parameters (generally those that are likely to be helpful in tuning and troubleshooting).

Note that parameter values shown here are those currently in effect (as opposed to the values shown in **mmlsconfig** output, which may show the settings that will become effective on the next GPFS restart).

--deadlock

Displays the longest waiters exceeding the deadlock detection thresholds.

If a deadlock situation occurs, administrators can use this information from all nodes in a cluster to help decide how to break up the deadlock.

--dmapi

Displays various DMAPI information. If no other options are specified, summary information is

mmdiag

displayed for sessions, pending events, cached tokens, stripe groups, and events waiting for reply. The `--dmapi` parameter accepts the following options:

session

Displays a list of sessions.

event

Displays a list of pending events.

token

Displays a list of cached tokens, stripe groups, and events waiting for reply.

disposition

Displays the DMAPI disposition for events.

all

Displays all of the **session**, **event**, **token**, and **disposition** information with additional details.

--iohist

Displays recent IO history. The information about IO requests recently submitted by GPFS code is shown here. It can provide some insight into various aspects of GPFS IO, such as the type of data or metadata being read or written, the distribution of IO sizes, and IO completion times for individual IOs. This information can be very useful in performance tuning and troubleshooting.

--lroc

Displays status and statistics for local read-only cache (LROC) devices.

--memory

Displays information about **mmfsd** memory usage. There are several distinct memory regions that **mmfsd** allocates and uses, and it can be important to know the memory usage situation for each one.

Heap memory allocated by mmfsd

This area is managed by the OS and does not have a preset limit enforced by GPFS.

Memory pools 1 and 2

Both of these refer to a single memory area, also known as the shared segment. It is used to cache various kinds of internal GPFS metadata, as well as for many other internal uses. This memory area is allocated using a special, platform-specific mechanism and is shared between user space and kernel code. The preset limit on the maximum shared segment size, current usage, and some prior usage information are shown here.

Memory pool 3

This area is also known as the token manager pool. This memory area is used to store the token state on token manager servers. The preset limit on the maximum memory pool size, current usage, and some prior-usage information are shown here.

This information can be useful when troubleshooting ENOMEM errors returned by GPFS to a user application, as well as memory allocation failures reported in a GPFS log file.

--network

Displays information about **mmfsd** network connections and pending Remote Procedure Calls (RPCs). Basic information and statistics about all existing **mmfsd** network connections to other nodes is displayed, including information about broken connections. If there are currently any RPCs pending (that is, sent but not yet replied to), the information about each one is shown, including the list of RPC destinations and the status of the request for each destination. This information can be very helpful in following a multinode chain of dependencies during a deadlock or performance-problem troubleshooting.

--rpc

Displays RPC performance statistics. The `--rpc` parameter accepts the following options:

node[=*name*]

Displays all per node statistics (channel wait, send time TCP, send time verbs, receive time TCP, latency TCP, latency verbs, and latency mixed). If *name* is specified, all per node statistics for just the specified node are displayed.

size

Displays per size range statistics.

message

Displays per message type RPC execution time.

all

Displays everything.

nn{*S*|*s*|*M*|*m*|*H*|*h*|*D*|*d*}

Displays per node RPC latency statistics for the latest number of intervals, specified by *nn*, for the interval specified by one of the following characters:

S*|*s

Displays second intervals only.

M*|*m

Displays first the second intervals since the last minute boundary followed by minute intervals.

H*|*h

Displays first the second and minute intervals since their last minute and hour boundary followed by hour intervals.

D*|*d

Displays first the second, minute, and hour intervals since their last minute, hour, and day boundary followed by day intervals.

Averages are displayed as a number of milliseconds with three decimal places (1 microsecond granularity).

--stats

Displays some general GPFS statistics.

GPFS uses a diverse array of objects to maintain the file system state and cache various types of metadata. The statistics about some of the more important object types are shown here.

OpenFile

This object is needed to access an inode. The target maximum number of cached OpenFile objects is governed by the **maxFilesToCache** configuration parameter. Note that more OpenFile objects may be cached, depending on workload.

CompactOpenFile

These objects contain an abbreviated form of an OpenFile, and are collectively known as *stat cache*. The target maximum number of cached CompactOpenFile objects is governed by the **maxStatCache** configuration parameter.

OpenInstance

This object is created for each open file instance (file or directory opened by a distinct process).

BufferDesc

This object is used to manage buffers in the GPFS pagepool.

indBlockDesc

This object is used to cache indirect block data.

All of these objects use the shared segment memory. For each object type, there is a preset target, derived from a combination of configuration parameters and the memory available in the shared segment. The information about current object usage can be helpful in performance tuning.

mmdiag

--threads

Displays **mmfsd** thread statistics and the list of active threads. For each thread, its type and kernel thread ID are shown. All non-idle **mmfsd** threads are shown. For those threads that are currently waiting for an event, the wait reason and wait time in seconds are shown. This information provides more detail than the data displayed by **mmdiag --waiters**.

--tokenmgr

Displays information about token management. For each mounted GPFS file system, one or more token manager nodes will be appointed. The first token manager is always colocated with the file system manager, while additional token managers may be appointed from the pool of nodes with the *manager* designation. The information shown here includes the list of currently appointed token manager nodes and, if the current node is serving as a token manager, some statistics about prior token transactions.

--trace

Display current trace status and trace levels. During GPFS troubleshooting, it is often necessary to use the trace subsystem to obtain the debug data necessary to understand the problem (see the topic about the GPFS trace facility in the *IBM Spectrum Scale: Problem Determination Guide*.) It is very important to have trace levels set correctly, per instructions provided by the IBM Support Center. The information shown here allows you to check the state of tracing and to see the trace levels currently in effect.

--version

Displays information about the GPFS build currently running on this node. This helps in troubleshooting installation problems. The information displayed here may be more comprehensive than version information available via the OS package management infrastructure, in particular when an e-fix is installed.

--waiters

Displays **mmfsd** threads waiting for events. This information can be very helpful in troubleshooting deadlocks and performance problems. For each thread, the thread name, wait time in seconds, and wait reason are typically shown. Only non-idle threads currently waiting for some event to occur are displayed. Note that only **mmfsd** threads are shown; any application IO threads that might be waiting in GPFS kernel code would not be present here.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmdiag** command.

Examples

1. To display a list of waiters, issue this command:

```
mmdiag --waiters
```

The system displays output similar to the following:

```
=== mmdiag: waiters ===
0x11DA520 waiting 0.001147000 seconds, InodePrefetchWorker:
  for I/O completion
0x2AAAAAB02830 waiting 0.002152000 seconds, InodePrefetchWorker:
  for I/O completion
0x2AAAAAB103990 waiting 0.000593000 seconds, InodePrefetchWorker:
  for I/O completion
0x11F51E0 waiting 0.000612000 seconds, InodePrefetchWorker:
```

```

for I/O completion
0x11EDE60 waiting 0.005736500 seconds, InodePrefetchWorker:
on ThMutex 0x100073ABC8 (0xFFFFC2000073ABC8)
(CacheReplacementListMutex)

```

In this example, all waiters have a very short wait duration and represent a typical snapshot of normal GPFS operation.

2. To display information about GPFS memory utilization, issue this command:

```
mmdiag --memory
```

The system displays output similar to the following:

```
mmfsd heap size: 1503232 bytes
```

```
current mmfsd heap bytes in use: 1919624 total 1867672 payload
```

```

Statistics for MemoryPool id 1 ("Shared Segment (EPHEMERAL)")
  128 bytes in use
 557721725 hard limit on memory usage
 1048576 bytes committed to regions
   1 allocations
   1 frees
   0 allocation failures

```

```

Statistics for MemoryPool id 2 ("Shared Segment")
 8355904 bytes in use
 557721725 hard limit on memory usage
 8785920 bytes committed to regions
1297534 allocations
1296595 frees
   0 allocation failures

```

```

Statistics for MemoryPool id 3 ("Token Manager")
 496184 bytes in use
 510027355 hard limit on memory usage
 524288 bytes committed to regions
 1309 allocations
  130 frees
   0 allocation failures

```

In this example, a typical memory usage picture is shown. None of the memory pools are close to being full, and there are no prior allocation failures.

Location

```
/usr/lpp/mmfs/bin
```

mmeditac1 command

Creates or changes a GPFS access control list.

Synopsis

```
mmeditac1 [-d] [-k {nfs4 | posix | native}] Filename
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmeditac1** command for interactive editing of the ACL of a file or directory. This command uses the default editor, specified in the EDITOR environment variable, to display the current access control information, and allows the file owner to change it. The command verifies the change request with the user before making permanent changes.

This command cannot be run from a Windows node.

The EDITOR environment variable must contain a complete path name, for example:

```
export EDITOR=/bin/vi
```

For information about NFS V4 ACLs, see the topics *Managing GPFS access control lists* and *NFS and GPFS in the IBM Spectrum Scale: Advanced Administration Guide*.

Users may need to see ACLs in their true form as well as how they are translated for access evaluations. There are four cases:

1. By default, **mmeditac1** returns the ACL in a format consistent with the file system setting, specified using the **-k** flag on the **mmcrfs** or **mmchfs** commands.
 - If the setting is **posix**, the ACL is shown as a traditional ACL.
 - If the setting is **nfs4**, the ACL is shown as an NFS V4 ACL.
 - If the setting is **all**, the ACL is returned in its true form.
2. The command **mmeditac1 -k nfs4** always produces an NFS V4 ACL.
3. The command **mmeditac1 -k posix** always produces a traditional ACL.
4. The command **mmeditac1 -k native** always shows the ACL in its true form regardless of the file system setting.

The following describes how **mmeditac1** works for POSIX and NFS V4 ACLs:

Command	ACL	mmcrfs -k	Display	-d (default)
mmeditac1	posix	posix	Access ACL	Default ACL
mmeditac1	posix	nfs4	NFS V4 ACL	Error[1]
mmeditac1	posix	all	Access ACL	Default ACL
mmeditac1	nfs4	posix	Access ACL[2]	Default ACL[2]
mmeditac1	nfs4	nfs4	NFS V4 ACL	Error[1]
mmeditac1	nfs4	all	NFS V4 ACL	Error[1]
mmeditac1 -k native	posix	any	Access ACL	Default ACL
mmeditac1 -k native	nfs4	any	NFS V4 ACL	Error[1]
mmeditac1 -k posix	posix	any	Access ACL	Default ACL
mmeditac1 -k posix	nfs4	any	Access ACL[2]	Default ACL[2]
mmeditac1 -k nfs4	any	any	NFS V4 ACL	Error[1]

[1] NFS V4 ACLs include inherited entries. Consequently, there cannot

be a separate default ACL.
 [2] Only the mode entries (owner, group, everyone) are translated.
 The **rwX** values are derived from the
 NFS V4 file mode attribute. Since the NFS V4 ACL is more granular
 in nature, some information is lost in this translation.

In the case of NFS V4 ACLs, there is no concept of a default ACL. Instead, there is a single ACL and the individual access control entries can be flagged as being inherited (either by files, directories, both, or neither). Consequently, specifying the **-d** flag for an NFS V4 ACL is an error. By its nature, storing an NFS V4 ACL implies changing the inheritable entries (the GPFS default ACL) as well.

Depending on the file system's **-k** setting (**posix**, **nfs4**, or **all**), **mmeditACL** may be restricted. The **mmeditACL** command is not allowed to store an NFS V4 ACL if **-k posix** is in effect, and is not allowed to store a POSIX ACL if **-k nfs4** is in effect. For more information, see the description of the **-k** flag for the **mmchfs**, **mmcrfs**, and **mmclsfs** commands.

Parameters

Filename

The path name of the file or directory for which the ACL is to be edited. If the **-d** option is specified, *Filename* must contain the name of a directory.

Options

-d Specifies that the default ACL of a directory is to be edited.

-k {nfs4 | posix | native}

nfs4

Always produces an NFS V4 ACL.

posix

Always produces a traditional ACL.

native

Always shows the ACL in its true form regardless of the file system setting.

This option should not be used for routine ACL manipulation. It is intended to provide a way to show the translations that are done. For example, if a **posix** ACL is translated by NFS V4. Beware that if the **-k nfs4** flag is used, but the file system does not allow NFS V4 ACLs, you will not be able to store the ACL that is returned. If the file system does support NFS V4 ACLs, the **-k nfs4** flag is an easy way to convert an existing **posix** ACL to **nfs4** format.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You may issue the **mmeditACL** command only from a node in the GPFS cluster where the file system is mounted.

The **mmeditACL** command may be used to display an ACL. POSIX ACLs may be displayed by any user with access to the file or directory. NFS V4 ACLs have a **READ_ACL** permission that is required for non-privileged users to be able to see an ACL. To change an existing ACL, the user must either be the owner, the root user, or someone with control permission (**WRITE_ACL** is required where the existing ACL is of type NFS V4).

mmeditac1

Examples

To edit the ACL for a file named **project2.history**, issue this command:

```
mmeditac1 project2.history
```

The current ACL entries are displayed using the default editor, provided that the EDITOR environment variable specifies a complete path name. When the file is saved, the system displays information similar to:

```
mmeditac1: 6027-967 Should the modified ACL be applied? (yes) or (no)
```

After responding **yes**, the ACLs are applied.

See also

- “mmdelacl command” on page 446
- “mmgetacl command” on page 500
- “mmputacl command” on page 614

Location

```
/usr/lpp/mmfs/bin
```

mmedquota command

Sets quota limits.

Synopsis

```
mmedquota {-u [-p [ProtoFileset:]ProtoUser] [Device:Fileset:]User ... |
           -g [-p [ProtoFileset:]ProtoGroup] [Device:Fileset:]Group ... |
           -j [-p ProtoFileset] Device:Fileset ... |
           -d {-u User ... | -g Group ... | -j Device:Fileset ...} |
           -t {{-u | -g | -j} [--reset]}}
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

The **mmedquota** command serves two purposes:

1. Sets or changes quota limits or grace periods for users, groups, and filesets in the cluster from which the command is issued.
2. Reestablishes user, group, or fileset default quotas for all file systems with default quotas enabled in the cluster.

The **mmedquota** command displays the current values for these limits, if any, and prompts you to enter new values using your default editor:

- current block usage (the amount of disk space used by this user, group, or fileset, in 1KB units; display only)
- current inode usage (display only)
- node soft limit
- inode hard limit
- block soft limit (the amount of disk space that this user, group, or fileset is allowed to use during normal operation)
Displayed in **KB**, but may be specified using **g**, **G**, **k**, **K**, **m**, **M**, **p**, **P**, **t**, or **T**. If no suffix is provided, the number is assumed to be in **bytes**.
- block hard limit (the total amount of disk space that this user, group, or fileset is allowed to use during the grace period)
Displayed in **KB**, but may be specified using **g**, **G**, **k**, **K**, **m**, **M**, **p**, **P**, **t**, or **T**. If no suffix is provided, the number is assumed to be in **bytes**.

Note: A block or inode limit of 0 indicates no limit.

The **mmedquota** command waits for the edit window to be closed before checking and applying new values. If an incorrect entry is made, reissue the command and enter the correct values.

You can also use the **mmedquota** command to change the file system-specific grace periods for block and file usage if the default of one week is unsatisfactory. The grace period is the time during which users can exceed the soft limit. If the user, group, or fileset does not show reduced usage below the soft limit before the grace period expires, the soft limit becomes the new hard limit.

When you set quota limits for a file system, consider replication in the file system. See the topic *Listing quotas* in the *IBM Spectrum Scale: Administration and Programming Reference*.

The EDITOR environment variable must contain a complete path name, for example:

```
export EDITOR=/bin/vi
```

mmedquota

Parameters

Device

Specifies the device name of the file system for which quota information is to be displayed. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

Fileset

Specifies the name of a fileset located on *Device* for which quota information is to be displayed.

User

Name or user ID of target user for quota editing.

Group

Name or group ID of target group for quota editing.

-d Reestablish default quota limits for a specific user, group, or fileset that has had an explicit quota limit set by a previous invocation of the **mmedquota** command.

-g Sets quota limits or grace times for groups.

-j Sets quota limits or grace times for filesets.

-p Applies already-established limits to a particular user, group or fileset.

When invoked with the **-u** option, [*ProtoFileset*:]*ProtoUser* limits are automatically applied to the specified *User* or space-delimited list of users.

When invoked with the **-g** option, [*ProtoFileset*:]*ProtoGroup* limits are automatically applied to the specified *Group* or space-delimited list of groups.

When invoked with the **-j** option, *ProtoFileset* limits are automatically applied to the specified fileset or space-delimited list of fileset names.

You can specify any user as a *ProtoUser* for another *User*, or any group as a *ProtoGroup* for another *Group*, or any fileset as a *ProtoFileset* for another *Fileset*.

-p cannot propagate a prototype quota from a user, group, or fileset on one file system to a user, group, or fileset on another file system.

-t Sets grace period during which quotas can exceed the soft limit before it is imposed as a hard limit. The default grace period is one week.

This flag is followed by one of the following flags: **-u**, **-g** or **-j**, to specify whether the changes apply to users, groups, or filesets respectively.

-u Sets quota limits or grace times for users.

--reset

With this option, when grace time is modified, all relative quota entries will be scanned and updated if necessary; without this option, when grace time is updated, quota entries will not be scanned and updated.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmedquota** command.

GPFS must be running on the node from which the **mmedquota** command is issued.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To set user quotas for userid **pfs001**, issue this command:

```
mmedquota -u pfs001
```

The system displays information similar to:

```
*** Edit quota limits for USR pfs001
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs3: (root): blocks in use: 8K, limits (soft = 0K, hard = 0K)
        inodes in use: 1, limits (soft = 0, hard = 0)
gpfs2: (fset4): blocks in use: 4104K, limits (soft = 102400K, hard = 153600K)
        inodes in use: 2, limits (soft = 100, hard = 150)
gpfs2: (fset3): blocks in use: 0K, limits (soft = 0K, hard = 0K)
        inodes in use: 0, limits (soft = 0, hard = 0)
gpfs2: (root): blocks in use: 0K, limits (soft = 0K, hard = 0K)
        inodes in use: 0, limits (soft = 0, hard = 0)
gpfs1: (fset1): blocks in use: 0K, limits (soft = 256K, hard = 256K)
        inodes in use: 0, limits (soft = 30, hard = 40)
gpfs1: (root): blocks in use: 0K, limits (soft = 256K, hard = 256K)
        inodes in use: 0, limits (soft = 40, hard = 45)
```

2. To reset default group quota values for the group **blueteam**, issue this command:

```
mmedquota -d -g blueteam
```

To verify the change, issue this command:

```
mmrepquota -q fs1
```

The system displays information similar to:

```
fs1: USR quota is on; default quota is on
fs1: GRP quota is on; default quota is on
fs1: FILESET quota is on; default quota is off
```

3. To change the grace periods for all users, issue this command:

```
mmedquota -t -u
```

The system displays information in your default editor similar to:

```
*** Edit grace times:
Time units may be : days, hours, minutes, or seconds
Grace period before enforcing soft limits for USRs:
gpfs0: block grace period: 7 days, file grace period: 7 days
```

4. To set user quotas for device **gpfs2**, fileset **fset3**, and userid **pfs001**, issue this command:

```
mmedquota -u gpfs2:fset3:pfs001
```

The system displays information similar to:

```
*** Edit quota limits for USR gpfs2:fset3:pfs001
NOTE: block limits will be rounded up to the next multiple of the block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs2: (fset3): blocks in use: 0K, limits (soft = 0K, hard = 0K)
        inodes in use: 0, limits (soft = 0, hard = 0)
```

5. To apply already-established limits of user **pfs002** to user **pfs001**, issue this command:

```
mmedquota -u -p pfs002 pfs001
```

The system displays information similar to:

mmedquota

Already established limits of protouser pfs002 in root fileset are applied to all filesets (including root fileset) in all corresponding per-fileset quota enabled filesystems.

6. To apply already-established limits of user **pfs002** in fileset **fset2** to user **pfs001** in fileset **fset1** and file system **fs1**, issue this command:

```
mmedquota -u -p fset2:pfs002 fs1:fset1:pfs001
```

The system displays information similar to:

```
Limits of protouser pfs002 in fileset fset2 in filesystem fs1 are applied  
to user pfs001 in fileset fset1
```

7. To apply an already-established fileset quota (from **fileset1** to **fileset2**) in two different file systems (**gpfstest1** and **gpfstest2**), issue this command:

```
mmedquota,-j -p fileset1 gpfstest1:fileset2 gpfstest2:fileset2
```

See also

- “mmcheckquota command” on page 361
- “mmdefedquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmlsquota command” on page 559
- “mmquotaon command” on page 619
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

```
/usr/lpp/mmfs/bin
```

mmexportfs command

Retrieves the information needed to move a file system to a different cluster.

Synopsis

```
mmexportfs {Device | all} -o ExportfsFile
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmexportfs** command, in conjunction with the **mmimportfs** command, can be used to move one or more GPFS file systems from one GPFS cluster to another GPFS cluster, or to temporarily remove file systems from the cluster and restore them at a later time. The **mmexportfs** command retrieves all relevant file system and disk information and stores it in the file specified with the **-o** parameter. This file must later be provided as input to the **mmimportfs** command. When running the **mmexportfs** command, the file system must be unmounted on all nodes.

When **all** is specified in place of a file system name, any disks that are not associated with a file system will be exported as well.

Exported file systems remain unusable until they are imported back with the **mmimportfs** command to the same or a different GPFS cluster.

Results

Upon successful completion of the **mmexportfs** command, all configuration information pertaining to the exported file system and its disks is removed from the configuration data of the current GPFS cluster and is stored in the user specified file *ExportfsFile*.

Parameters

Device | **all**

The device name of the file system to be exported. File system names need not be fully-qualified. *fs0* is as acceptable as */dev/fs0*. Specify **all** to export all GPFS file systems, as well as all disks that do not currently belong to a file system.

If the specified file system device is a GPFS Native RAID-based file system, then all affected GPFS Native RAID objects will be exported as well. This includes recovery groups, declustered arrays, vdisks, and any other file systems that are based on these objects. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

This must be the first parameter.

-o *ExportfsFile*

The path name of a file to which the file system information is to be written. This file must be provided as input to the subsequent **mmimportfs** command.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmexportfs

Security

You must have root authority to run the **mmexportfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To export all file systems in the current cluster, issue this command:

```
mmexportfs all -o /u/admin/exportfile
```

The output is similar to this:

```
mmexportfs: Processing file system fs1 ...
```

```
mmexportfs: Processing file system fs2 ...
```

```
mmexportfs: Processing disks that do not belong to any file system ...
```

```
mmexportfs: 6027-1371 Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

See also

- “mmimportfs command” on page 513

Location

```
/usr/lpp/mmfs/bin
```

mmfsck command

Checks and repairs a GPFS file system.

Synopsis

```
mmfsck Device [-n | -y] [-s | -v | -V]
             [-c | -m | -o | --skip-inode-check | --skip-directory-check]
             [-t Directory]
             [-N {Node[,Node...] | NodeFile | NodeClass}]
             [--patch-file Path [--patch]] [--qos QosClass]
             [--threads ThreadLevel]
```

The file system must be unmounted before you can run the **mmfsck** command with any option other than **-o**.

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmfsck** command in offline mode is intended to be used only in situations where there have been disk or communications failures that have caused **MMFS_FSSTRUCT** error log entries to be issued, or where it is known that disks have been forcibly removed or otherwise permanently unavailable for use in the file system, and other unexpected symptoms are seen by users. In general it is unnecessary to run **mmfsck** in offline mode unless under the direction of the IBM Support Center.

If neither the **-n** nor **-y** flag is specified, the **mmfsck** command runs interactively prompting you for permission to repair each consistency error as reported. It is suggested that in all but the most severely damaged file systems, you run the **mmfsck** command interactively (the default).

The occurrence of I/O errors, or the appearance of a message telling you to run the **mmfsck** command, may indicate file system inconsistencies. If either situation occurs, use the **mmfsck** command to check file system consistency and interactively repair the file system.

For information about file system maintenance and repair, see the topic *Checking and repairing a file system* in the *IBM Spectrum Scale: Advanced Administration Guide*. The **mmfsck** command checks for these inconsistencies:

- Blocks marked allocated that do not belong to any file. The corrective action is to mark the block free in the allocation map.
- Files for which an inode is allocated and no directory entry exists (orphaned files). The corrective action is to create directory entries for these files in a **lost+found** subdirectory of the fileset to which the orphaned file or directory belongs. The index number of the inode is assigned as the name. If you do not allow the **mmfsck** command to reattach an orphaned file, it asks for permission to delete the file.
- Directory entries pointing to an inode that is not allocated. The corrective action is to remove the directory entry.
- Incorrectly formed directory entries. A directory file contains the inode number and the generation number of the file to which it refers. When the generation number in the directory does not match the generation number stored in the file's inode, the corrective action is to remove the directory entry.
- Incorrect link counts on files and directories. The corrective action is to update them with accurate counts.
- Policy files are not valid. The corrective action is to delete the file.

mmfsck

- Various problems related to filesets: missing or corrupted fileset metadata, inconsistencies in directory structure related to filesets, missing or corrupted fileset root directory, other problems in internal data structures. The repaired filesets will be renamed as **FilesetFilesetId** and put into unlinked state.

If you are repairing a file system due to node failure and the file system has quotas enabled, it is suggested that you run the **mmcheckquota** command to recreate the quota files.

Indications leading you to the conclusion that you should run the **mmfsck** command include:

- An **MMFS_FSSTRUCT** along with an **MMFS_SYSTEM_UNMOUNT** error log entry on any node indicating some critical piece of the file system is inconsistent.
- Disk media failures
- Partial disk failure
- **EVALIDATE=214**, Invalid checksum or other consistency check failure on a disk data structure, reported in error logs or returned to an application.

For further information on recovery actions and how to contact the IBM Support Center, see the *IBM Spectrum Scale: Problem Determination Guide*.

If you are running the online **mmfsck** command to free allocated blocks that do not belong to any files, plan to make file system repairs when system demand is low. This is an I/O intensive activity and it can affect system performance.

Results

If the file system is inconsistent, the **mmfsck** command displays information about the inconsistencies and (depending on the option entered) may prompt you for permission to repair them. The **mmfsck** command tries to avoid actions that may result in loss of data. In some cases, however, it may indicate the destruction of a damaged file.

All corrective actions, with the exception of recovering lost disk blocks (blocks that are marked as allocated but do not belong to any file), require that the file system be unmounted on all nodes. If the **mmfsck** command is run on a mounted file system, lost blocks are recovered but any other inconsistencies are only reported, not repaired.

If a bad disk is detected, the **mmfsck** command stops the disk and writes an entry to the error log. The operator must manually start and resume the disk when the problem is fixed.

The file system must be unmounted on all nodes before the **mmfsck** command can repair file system inconsistencies.

Parameters

Device

The device name of the file system to be checked and repaired. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

- n Specifies a **no** response to all file system error repair prompts from the **mmfsck** command. The option reports inconsistencies but it does not change the file system. To save this information, redirect it to an output file when you issue the **mmfsck** command.
- y Specifies a **yes** response to all file system error repair prompts from the **mmfsck** command. Use this option only on severely damaged file systems. It allows the **mmfsck** command to take any action necessary for repairs.
- s Specifies that the output is semi-verbose.

- v Specifies that the output is verbose.
- V Specifies that the output is verbose and contains information for debugging purposes.
- c When the file system log has been lost and the file system is replicated, this option specifies that the **mmfsck** command attempt corrective action by comparing the replicas of metadata and data. If this error condition occurs, it is indicated by an error log entry.
- m Has the same meaning as **-c**, except that **mmfsck** checks only the metadata replica blocks. It therefore runs faster than with **-c**.
- o Online mode does not perform a full file system consistency check, but blocks marked as allocated that do not belong to a file are recovered. Lost blocks do not constitute file system corruption.

--skip-inode-check

Causes the command to run faster by skipping its inode-check phase. Include this option only if you know that the inodes are valid and that only directories need to be checked. In this mode, the product does not scan all parts of the file system and therefore might not detect all corruptions in the file system.

--skip-directory-check

Causes the command to run faster by skipping its directory-check phase. Include this option if you want to check only the inodes. In this mode, the product does not scan all parts of the file system and therefore might not detect all corruptions in the file system.

-t *Directory*

Specifies the directory that GPFS uses for temporary storage during **mmfsck** command processing. This directory must be available on all nodes that are participating in **mmfsck** and that are designated as either manager or quorum node. In addition to the location requirement, the storage directory has a minimum space requirement of 4GB. The default directory for **mmfsck** processing is `/tmp`.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specify the nodes to participate in the check and repair of the file system. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For information on how to specify node names, see the topic *Specifying nodes as inputs to GPFS commands* in the *IBM Spectrum Scale: Advanced Administration Guide*.

--patch-file *Path*

Specifies the name of a patch file. When the **--patch** parameter is not specified, information about file system inconsistencies (detected during an **mmfsck** run with the **-n** parameter) are stored in the patch file that is specified by *Path*. *Path* should be accessible from the file system manager node. The information stored in the patch file can be viewed as a report of the problems in the file system. For more information about patch files, see the topic *Checking and repairing a file system* in the *IBM Spectrum Scale: Advanced Administration Guide*.

When this option is specified with the **--patch** parameter, the information in the patch file is read and used to repair the file system.

--patch

Specifies that the file system will be repaired using the information stored in the patch file that is specified with **--patch-file** *Path*.

--qos *QoSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

mmfsck

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

--threads *ThreadLevel*

The number of threads that are created to run **mmfsck**. The default is 16.

Exit status

- 0 Successful completion.
- 2 The command was interrupted before it completed checks or repairs.
- 4 The command changed the file system and it must now be restarted.
- 8 The file system contains damage that has not been repaired.
- 16 The problem cannot be fixed.

The exit string is a combination of three different error indicators:

1. The first value is the Exit **errno** value.
2. The second value is an internal ancillary value that helps explain where the **errno** value came from.
3. The third value is the OR of several status bits.

Security

You must have root authority to run the **mmfsck** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To run the **mmfsck** command on the **fs1** file system, receive a report, but not fix inconsistencies, issue this command:

```
mmfsck fs1 -n
```

The system displays information similar to:

```
Checking "fs1"  
Checking reserved files  
Checking inodes  
Checking inode map file  
Checking ACL file records  
Checking directories and files  
Checking log files  
Checking extended attributes file  
Checking allocation summary file  
Checking policy file  
Checking metadata of filesets  
Checking file reference counts  
Checking file system replication status
```



```

500224 inodes
   39 allocated
   0  repairable
   0  repaired
   0  damaged
   0  deallocated
   0  orphaned
   0  attached
   0  corrupt ACL references

13107200 subblocks
 142696 allocated
   0  unreferenced
   0  duplicates
   0  deletable
   0  deallocated

4209 addresses
   0  suspended
   0  duplicates
   0  reserved file holes found
   0  reserved file holes repaired
File system is clean.

```

mmfsck found no inconsistencies in this file system.

- To run the **mmfsck** command on the **fs2** file system, receive a report, and fix inconsistencies, issue this command:

```
mmfsck fs2 -y
```

The system displays information similar to:

```
Checking "fs2"
Checking inodes
```

```
Lost blocks were found.
Correct the allocation map? yes
Checking inode map file
```

```
Corrections are needed in the inode allocation map.
Correct the allocation map? yes
Root inode 32512 of fileset 'fset2' has been deleted.
Delete the inode reference from fileset metadata? yes
Checking directories and files
```

```
Error in directory inode 3: DirEntryBad DirLinkCountBad
Directory entry "top_dir" is not an allocated inode.
Patching will delete the directory entry.
Remove directory entry? yes
Directory entry "fset2" is not an allocated inode.
Patching will delete the directory entry.
Remove directory entry? yes
Directory has an incorrect link count of 4.
Corrected link count would be 2
Correct link count? yes
```

```
Error in directory inode 12032: DirEntryBad
Directory entry "." is not an allocated inode.
Cannot allow deletion of this directory entry.
```

```
Error in directory inode 12034: DirEntryBad BadFilesetId
Directory entry "." has a fileset id that does not match fileset id of the directory.
Patching will reset fileset id of inode 32512 to the fileset id of the directory, 1
Correct fileset id? yes
Directory entry "." is not an allocated inode.
Cannot allow deletion of this directory entry.
```

mmfsck

```
Checking log files
Checking extended attributes file
Checking allocation summary file
Checking policy file
Checking filesets metadata
Root directory of fileset 'fset2' (inode -1) is invalid
Recreate fileset root inode and directory? yes
Checking file reference counts

Directory inode 12032 is not referenced in any directory.
Reattach inode to lost+found? yes

Directory inode 12034 is not referenced in any directory.
Reattach inode to lost+found? yes
Checking file system replication status
  10585856 inodes
    369 allocated
    42 repairable
    42 repaired
    0 damaged
    0 deallocated
    0 orphaned
    0 attached
    0 corrupt ACL references

  89391104 subblocks
    661908 allocated
    262 unreferenced
    0 duplicates
    0 deletable
    262 deallocated

  20464 addresses
    0 suspended
    0 duplicates
    0 reserved file holes found
    0 reserved file holes repaired

File system is clean.
```

3. To run the **mmfsck** command on the **FSchk** file system, and create a patch file called **path-towrite-patchfile** that will store information about the file system inconsistencies, issue this command:

```
mmfsck FSchk -nv --patch-file path-towrite-patchfile
```

The system displays information similar to:

```
Creating patch file "path-towrite-patchfile" on node "Node3"
Checking "FSchk"
  fsckFlags          0x8000009
  Stripe group manager <c0n3>
  needNewLogs        0
...

Checking inode map segment 0 of 1
Inode 329478 not in use but marked (0x0).

Corrections are needed in the inode allocation map.
Correct the allocation map? No
  Checking inode map segment 1 of 1
Checking inode map for inode range 388608 to 518143
  Checking inode map segment 0 of 1
  Checking inode map segment 1 of 1
Checking inode map for inode range 518144 to 624383
  Checking inode map segment 0 of 1
  Checking inode map segment 1 of 1
Inode 527110 not in use but marked (0x0).
2 inodes are not in use but marked.
```

```

Error in directory inode 329477: DirEntryBad DirLinkCountBad
Directory entry "dir2" is not an allocated inode.
Patching will delete the directory entry.
Remove directory entry? No
Directory has an incorrect link count of 3.
Corrected link count would be 2
Correct link count? No

```

```

Error in directory inode 527104: DirEntryBad DirLinkCountBad
Directory entry "dir_2" is not an allocated inode.
Patching will delete the directory entry.
Remove directory entry? No
Directory has an incorrect link count of 3.
Corrected link count would be 2
Correct link count? No

```

...

```

Error in inode 527109: SubblocksBad
Inode 527109 has an incorrect subblock count of 40.
Corrected subblock count would be 32.
Correct count? No

```

```

Error in inode 329474: SubblocksBad
Inode 329474 has an incorrect subblock count of 37.
Corrected subblock count would be 32.
Correct count? No

```

```

Error in inode 329480: RepCountBad
Inode 329480 has an incorrect current metadata replica count of 3.
Correct replication count? No

```

...

```

Scanning directories for cycle
Checking log files
Checking extended attributes file
Checking allocation summary file
Checking policy file
Checking metadata of filesets
Checking file reference counts

```

...

```

624384 inodes
    67 allocated
    5 repairable
    0 repaired
    0 damaged
    0 deallocated
    5 orphaned
    0 attached
    0 corrupt ACL references

2234776 subblocks
    191237 allocated
    0 unreferenced
    0 duplicates
    0 deletable
    0 deallocated

5860 addresses
    0 suspended
    0 duplicates
    0 reserved file holes found

```

mmfsck

0 reserved file holes repaired

InodeProblemList: 5 entries

iNum	snapId	status	keep	delete	noScan	new	error
527109	0	1	0	0	0	0	0x00080000 SubblocksBad
329474	0	1	0	0	0	0	0x00080000 SubblocksBad
329480	0	1	0	0	0	0	0x00000800 RepCountBad
329477	0	1	0	0	0	0	0x00009000 DirEntryBad DirLinkCountBad
527104	0	1	0	0	0	0	0x00009000 DirEntryBad DirLinkCountBad

File system contains unrepaired damage.

Exit status 0:0:8.

Patch file written to "Node3:path-towrite-patchfile" with 13 patch entries.

mmfsck: 6027-1639 Command failed. Examine previous error messages to determine cause.

To use the information that was stored in the patch file (**path-towrite-patchfile**) to repair the file system, issue the following command:

```
mmfsck FSchk -v --patch-file path-towrite-patchfile --patch
```

The system displays information similar to:

Checking "FSchk"

```
fsckFlags          0x18000008
Stripe group manager <c0n3>
needNewLogs        0
nThreads           16
committed nodes    0
clientTerm         0
fsckReady          1
fsckCreated        0
% pool allowed     50
```

```
checkFilesets      1
```

```
checkFilesetsV2    1
```

Checking patch file

Scanning patch file for allocation map patches

Patching inode_map block 7 in inode 2 snap 0

Patching segment 234 index 6 from 0 to 3

Patching inode_map block 8 in inode 2 snap 0

Patching segment 259 index 6 from 0 to 3

Completed patching 2 allocation map patches.

52 % complete on Thu Apr 30 08:03:24 2015

Scanning patch file for inode patches

Patching inode block 2574 in inode 0 snap 0

Patching record 329474 field inode_last_block_subblocks from 37 to 32

Patching record 329480 field inode_curr_meta_replicas from 3 to 2

Patching inode block 4118 in inode 0 snap 0

Patching record 527109 field inode_last_block_subblocks from 40 to 32

Patching directory block 0 in inode 527104 snap 0

Deleting directory entry dir_2 at offset 352

Patching inode block 4118 in inode 0 snap 0

Patching record 527104 field inode_num_links from 3 to 2

Patching directory block 0 in inode 329477 snap 0

Deleting directory entry dir2 at offset 64

Patching inode block 2574 in inode 0 snap 0

Patching record 329477 field inode_num_links from 3 to 2

Orphaning inode 329483 snap 0

Orphaning inode 329484 snap 0

Orphaning inode 329485 snap 0

Orphaning inode 329486 snap 0

Orphaning inode 527113 snap 0

Completed patching 12 inode patches.

100 % complete on Thu Apr 30 08:03:24 2015

File system is clean.

See also

- “mmcheckquota command” on page 361
- “mmcrfs command” on page 414
- “mmdelfs command” on page 458
- “mmdf command” on page 470
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmfsctl

mmfsctl command

Issues a file system control request.

Synopsis

```
mmfsctl Device {suspend | suspend-write | resume}
```

or

```
mmfsctl Device {exclude | include}
               {-d "DiskName[;DiskName...]" | -F DiskFile | -G FailureGroup}
```

or

```
mmfsctl Device syncFSconfig
               {-n RemoteNodesFile | -C RemoteClusterName} [-S SpecFile]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmfsctl** command to issue control requests to a particular GPFS file system. The command is used to temporarily suspend the processing of all application I/O requests, and later resume them, as well as to synchronize the file system's configuration state between peer clusters in disaster recovery environments.

See “Establishing disaster recovery for your GPFS cluster” in *IBM Spectrum Scale: Advanced Administration Guide*.

Using **mmfsctl suspend** and **mmfsctl resume**

Before creating a FlashCopy® image of the file system, the user must run **mmfsctl suspend** to temporarily quiesce all file system activity and flush the internal buffers on all nodes that mount this file system. The on-disk metadata will be brought to a consistent state, which provides for the integrity of the FlashCopy snapshot. If a request to the file system is issued by the application after the invocation of this command, GPFS suspends this request indefinitely, or until the user issues **mmfsctl resume**.

Once the FlashCopy image has been taken, the **mmfsctl resume** command can be issued to resume the normal operation and complete any pending I/O requests.

Using **mmfsctl syncFSconfig**

The **mmfsctl syncFSconfig** command extracts the file system's related information from the local GPFS configuration data, transfers this data to one of the nodes in the peer cluster, and attempts to import it there.

Once the GPFS file system has been defined in the primary cluster, users run this command to import the configuration of this file system into the peer recovery cluster. After producing a FlashCopy image of the file system and propagating it to the peer cluster using Peer-to-Peer Remote Copy (PPRC), users similarly run this command to propagate any relevant configuration changes made in the cluster after the previous snapshot.

The primary cluster configuration server of the peer cluster must be available and accessible using remote shell and remote copy at the time of the invocation of the **mmfsctl syncFSconfig** command, and remote nodes must be reachable by the ping utility. Also, the peer GPFS clusters should be defined to use the same remote shell and remote copy mechanism, and they must be set up to allow nodes in peer clusters to communicate without the use of a password.

Note: In a cluster that is CCR-enabled, you cannot run **mmfsctl syncFSconfig** on a file system that has tiebreaker disks.

Not all administrative actions performed on the file system necessitate this type of resynchronization. It is required only for those actions that modify the file system information maintained in the local GPFS configuration data. These actions include:

- Adding, removing, and replacing disks (commands **mmadddisk**, **mmdeldisk**, **mmrpldisk**)
- Modifying disk attributes (command **mmchdisk**)
- Changing the file system's mount point (command **mmchfs -T**)
- Changing the file system device name (command **mmchfs -W**)

The process of synchronizing the file system configuration data can be automated by utilizing the **syncfsconfig** user exit.

Using **mmfsctl exclude**

The **mmfsctl exclude** command is to be used only in a disaster recovery environment, only after a disaster has occurred, and only after ensuring that the disks in question have been physically disconnected. Otherwise, unexpected results may occur.

The **mmfsctl exclude** command can be used to manually override the file system descriptor quorum after a site-wide disaster. See “Establishing disaster recovery for your GPFS cluster” in *IBM Spectrum Scale: Advanced Administration Guide*. This command enables users to restore normal access to the file system with less than a quorum of available file system descriptor replica disks, by effectively excluding the specified disks from all subsequent operations on the file system descriptor. After repairing the disks, the **mmfsctl include** command can be issued to restore the initial quorum configuration.

Parameters

Device

The device name of the file system. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**. If **all** is specified with the **syncFSconfig** option, this command is performed on all GPFS file systems defined in the cluster.

The following options can be specified after *Device*:

suspend

Instructs GPFS to flush the internal buffers on all nodes, bring the file system to a consistent state on disk, and suspend the processing of all subsequent application I/O requests.

suspend-write

Suspends the execution of all new write I/O requests coming from user applications, flushes all pending requests on all nodes, and brings the file system to a consistent state on disk.

resume

Instructs GPFS to resume the normal processing of I/O requests on all nodes.

exclude

Instructs GPFS to exclude the specified group of disks from all subsequent operations on the file system descriptor, and change their availability state to **down**, if the conditions in the following Note are met.

If necessary, this command assigns additional disks to serve as the disk descriptor replica holders, and migrate the disk descriptor to the new replica set. The excluded disks are not deleted from the file system, and still appear in the output of the **mmlsdisk** command.

Note: The **mmfsctl exclude** command is to be used only in a disaster recovery environment, only after a disaster has occurred, and only after ensuring that the disks in question have been physically disconnected. Otherwise, unexpected results may occur.

mmfsctl

include

Informs GPFS that the previously excluded disks have become operational again. This command writes the up-to-date version of the disk descriptor to each of the specified disks, and clears the **excl** tag.

-d "*DiskName* [*;DiskName...*]"

Specifies the names of the NSDs to be included or excluded by the **mmfsctl** command. Separate the names with semicolons (;) and enclose the list of disk names in quotation marks.

-F *DiskFile*

Specifies a file containing the names of the NSDs, one per line, to be included or excluded by the **mmfsctl** command.

-G *FailureGroup*

A failure group identifier for the disks to be included or excluded by the **mmfsctl** command.

syncFSconfig

Synchronizes the configuration state of a GPFS file system between the local cluster and its peer in two-cluster disaster recovery configurations.

The following options can be specified after **syncFSconfig**:

-n *RemoteNodesFile*

Specifies a list of contact nodes in the peer recovery cluster that GPFS uses when importing the configuration data into that cluster. Although any node in the peer cluster can be specified here, users are advised to specify the identities of the peer cluster's primary and secondary cluster configuration servers, for efficiency reasons.

-C *RemoteClusterName*

Specifies the name of the GPFS cluster that owns the remote GPFS file system.

-S *SpecFile*

Specifies the description of changes to be made to the file system, in the peer cluster during the import step. The format of this file is identical to that of the *ChangeSpecFile* used as input to the **mmimportfs** command. This option can be used, for example, to define the assignment of the NSD servers for use in the peer cluster.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Results

The **mmfsctl** command returns 0 if successful.

Security

You must have root authority to run the **mmfsctl** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

This sequence of commands creates a FlashCopy image of the file system and propagates this image to the recovery cluster using the Peer-to-Peer Remote Copy technology. The following configuration is assumed:

Site	LUNs
Primary cluster (site A)	lunA1, lunA2
Recovery cluster (site B)	lunB1

lunA1 FlashCopy source

lunA2 FlashCopy target, PPRC source

lunB1 PPRC target

A single GPFS file system named **fs0** has been defined in the primary cluster over lunA1.

1. In the primary cluster, suspend all file system I/O activity and flush the GPFS buffers

```
mmfsctl fs0 suspend
```

The output is similar to this:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
```

2. Establish a FlashCopy pair using lunA1 as the source and lunA2 as the target.

3. Resume the file system I/O activity:

```
mmfsctl fs0 resume
```

The output is similar to this:

```
Resuming operations.
```

4. Establish a Peer-to-Peer Remote Copy (PPRC) path and a synchronous PPRC volume pair lunA2-lunB1 (primary-secondary). Use the 'copy entire volume' option and leave the 'permit read from secondary' option disabled.
5. Wait for the completion of the FlashCopy background task. Wait for the PPRC pair to reach the duplex (fully synchronized) state.
6. Terminate the PPRC volume pair lunA2-lunB1.
7. If this is the first time the snapshot is taken, or if the configuration state of **fs0** changed since the previous FlashCopy snapshot, propagate the most recent configuration to site B:

```
mmfsctl fs0 syncFSconfig -n recovery_clust_nodelist
```

Location

```
/usr/lpp/mmfs/bin
```

mmgetacl

mmgetacl command

Displays the GPFS access control list of a file or directory.

Synopsis

```
mmgetacl [-d] [-o OutFilename] [-k {nfs4 | posix | native}] Filename
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmgetacl** command to display the ACL of a file or directory.

For information about NFS V4 ACLs, see the topics *Managing GPFS access control lists* and *NFS and GPFS in the IBM Spectrum Scale: Advanced Administration Guide*.

Users may need to see ACLs in their true form as well as how they are translated for access evaluations. There are four cases:

1. By default, **mmgetacl** returns the ACL in a format consistent with the file system setting, specified using the **-k** flag on the **mmcrfs** or **mmchfs** commands.
If the setting is **posix**, the ACL is shown as a traditional ACL.
If the setting is **nfs4**, the ACL is shown as an NFS V4 ACL.
If the setting is **all**, the ACL is returned in its true form.
2. The command **mmgetacl -k nfs4** always produces an NFS V4 ACL.
3. The command **mmgetacl -k posix** always produces a traditional ACL.
4. The command **mmgetacl -k native** always shows the ACL in its true form regardless of the file system setting.

The following describes how **mmgetacl** works for POSIX and NFS V4 ACLs:

Command	ACL	mmcrfs -k	Display	-d (default)
mmgetacl	posix	posix	Access ACL	Default ACL
mmgetacl	posix	nfs4	NFS V4 ACL	Error[1]
mmgetacl	posix	all	Access ACL	Default ACL
mmgetacl	nfs4	posix	Access ACL[2]	Default ACL[2]
mmgetacl	nfs4	nfs4	NFS V4 ACL	Error[1]
mmgetacl	nfs4	all	NFS V4 ACL	Error[1]
mmgetacl -k native	posix	any	Access ACL	Default ACL
mmgetacl -k native	nfs4	any	NFS V4 ACL	Error[1]
mmgetacl -k posix	posix	any	Access ACL	Default ACL
mmgetacl -k posix	nfs4	any	Access ACL[2]	Default ACL[2]
mmgetacl -k nfs4	any	any	NFS V4 ACL	Error[1]

[1] NFS V4 ACLs include inherited entries. Consequently, there cannot be a separate default ACL.

[2] Only the mode entries (owner, group, everyone) are translated. The **rwX** values are derived from the NFS V4 file mode attribute. Since the NFS V4 ACL is more granular in nature, some information is lost in this translation.

Parameters

Filename

The path name of the file or directory for which the ACL is to be displayed. If the **-d** option is specified, *Filename* must contain the name of a directory.

Options

-d Specifies that the default ACL of a directory is to be displayed.

-k {nfs4 | posix | native}

nfs4

Always produces an NFS V4 ACL.

posix

Always produces a traditional ACL.

native

Always shows the ACL in its true form regardless of the file system setting.

-o OutFilename

The path name of a file to which the ACL is to be written.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have read access to the directory where the file exists to run the **mmgetacl** command.

You may issue the **mmgetacl** command only from a node in the GPFS cluster where the file system is mounted.

Examples

1. To display the ACL for a file named **project2.history**, issue this command:

```
mmgetacl project2.history
```

The system displays information similar to:

```
#owner:paul
#group:design
user::rwx
group::r-x-
other::r-x-
```

2. This is an example of an NFS V4 ACL displayed using **mmgetacl**. Each entry consists of three lines reflecting the greater number of permissions in a text format. An entry is either an **allow** entry or a **deny** entry. An **X** indicates that the particular permission is selected, a minus sign (-) indicates that is it not selected. The following access control entry explicitly allows **READ**, **EXECUTE** and **READ_ATTR** to the **staff** group on a file:

```
group:staff:r-x-:allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (X)READ_ATTR (-)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

3. This is an example of a directory ACLs, which may include *inherit* entries (the equivalent of a default ACL). These do not apply to the directory itself, but instead become the initial ACL for any objects created within the directory. The following access control entry explicitly denies **READ/LIST**, **READ_ATTR**, and **EXEC/SEARCH** to the **sys** group.

```
group:sys:----:deny:DirInherit
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL (X)READ_ATTR (-)READ_NAMED
(-)DELETE (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

mmgetacl

See also

- “mmeditACL command” on page 478
- “mmdelACL command” on page 446
- “mmputACL command” on page 614

Location

/usr/lpp/mmfs/bin

mmgetstate command

Displays the state of the GPFS daemon on one or more nodes.

Synopsis

```
mmgetstate [-L] [-s] [-v] [-a | -N {Node[,Node...] | NodeFile | NodeClass}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmgetstate** command to show the state of the GPFS daemon on one or more nodes.

Parameters

-a Show the state of the GPFS daemon on all nodes in the cluster.

-N {Node[,Node...] | NodeFile | NodeClass}

Directs the **mmgetstate** command to return GPFS daemon information for a set of nodes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

Options

-L Display quorum, number of nodes up, total number of nodes, and other extended node information.

-s Display summary information such as: number of local and remote nodes that have joined in the cluster, number of quorum nodes.

-v Display intermediate error messages.

The GPFS states recognized and displayed by this command are:

active GPFS is ready for operations.

arbitrating

A node is trying to form a quorum with the other available nodes.

down GPFS daemon is not running on the node or is recovering from an internal error.

unknown

Unknown value. Node cannot be reached or some other error occurred.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmgetstate** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For

mmgetstate

more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To display the quorum, the number of nodes up, and the total number of nodes for the GPFS cluster, issue:

```
mmgetstate -a -L
```

The system displays output similar to:

Node number	Node name	Quorum	Nodes up	Total nodes	GPFS state	Remarks
1	c5n92	3	5	12	active	
2	c5n94	3	5	12	active	
3	c5n95	3	5	12	active	quorum node
4	c5n96	3	5	12	active	
5	c5n97	3	5	12	active	quorum node
6	c5n98	3	5	12	active	
7	c5n107	3	5	12	active	quorum node
8	c5n108	3	5	12	active	
9	c5n109	3	5	12	active	quorum node
10	c5n110	3	5	12	down	
11	c5n111	3	5	12	active	quorum node
12	c5n112	3	5	12	active	

The 3 under the Quorum column means that you must have three quorum nodes up to achieve quorum.

2. This is an example of a cluster using node quorum with tiebreaker disks. Note the * in the Quorum field, which indicates that tiebreaker disks are being used:

```
mmgetstate -a -L
```

The system displays output similar to:

Node number	Node name	Quorum	Nodes up	Total nodes	GPFS state	Remarks
1	k5n91	5*	8	21	active	
2	k5n92	5*	8	21	active	quorum node
3	k5n94	5*	8	21	active	
5	k5n96	5*	8	21	active	
6	k5n97	5*	8	21	active	quorum node
7	k5n98	5*	8	21	active	
8	k5n99	5*	8	21	active	quorum node

3. To display summary information, issue this command:

```
mmgetstate -s
```

The system displays output similar to:

Node number	Node name	GPFS state
5	c5n97	active

Summary information

```
-----  
Number of nodes defined in the cluster:      12  
Number of local nodes active in the cluster:  12  
Number of remote nodes joined in this cluster: 0  
Number of quorum nodes defined in the cluster: 5  
Number of quorum nodes active in the cluster: 5  
Quorum = 3, Quorum achieved
```

See also

- “mmchconfig command” on page 331
- “mmcrcluster command” on page 403

- “mmshutdown command” on page 661
- “mmstartup command” on page 678

Location

/usr/lpp/mmfs/bin

mmhadoopctl command

Installs and sets up the GPFS connector for a Hadoop distribution; starts or stops the GPFS connector daemon on a node.

Synopsis

```
mmhadoopctl connector {start | stop | getstate}
                    [-N {Node[,Node...] | NodeFile | NodeClass}]
```

or

```
mmhadoopctl connector {attach | detach}
                    --distribution HadoopDistribution
                    [-N {Node[,Node...] | NodeFile | NodeClass}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmhadoopctl** command to install and set up the GPFS connector for a Hadoop distribution, or to start or stop the GPFS connector daemon on a node.

Parameters

connector

Controls the GPFS connector daemon with one of the following actions:

start

Starts the connector daemon.

stop

Stops the connector daemon.

getstate

Detects whether the connector daemon is running and shows its process ID.

attach

Installs the GPFS connector daemon and required libraries into the designated Hadoop distribution.

detach

Uninstalls the GPFS connector daemon and required libraries from the designated Hadoop distribution.

--distribution *HadoopDistribution*

Designates a Hadoop distribution. Currently, the following values are accepted:

BigInsights

Apache

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the node or nodes on which the command is to be run (if you want to run the command not only on the local host but also on one or more other nodes). If you do not specify this option, the command is run on the local host only.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmhadoopctl** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To start the GPFS connector daemon, issue this command:

```
mmhadoopctl connector start
```

The system displays output similar to this:

```
Hadoop connector 'gpfs-connector-daemon' started
```

Location

```
/usr/lpp/mmfs/bin
```

mmimgbackup command

Performs a backup of a single GPFS file system metadata image.

Synopsis

```
mmimgbackup Device [-g GlobalWorkDirectory]
                 [-L n] [-N {Node[,Node...] | NodeFile | NodeClass}]
                 [-S SnapshotName] [--image ImageSetName] [--notsm | --tsm]
                 [--tsm-server ServerName] [POLICY-OPTIONS]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmimgbackup** command performs a backup of a single GPFS file system metadata image.

You must run the **mmbackupconfig** command before you run the **mmimgbackup** command. For more information, see the topic *Scale Out Backup and Restore (SOBAR)* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

Device

The device name of the file system whose metadata image is to be backed up. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-g *GlobalWorkDirectory*

The directory to be used for temporary files that need to be shared between the **mmimgbackup** worker nodes and to hold backup images until sent to archive. The default is:

```
mount_point_for_Device/.mmimgbackup
```

-L *n*

Controls the level of information displayed by the **mmimgbackup** command. The default for **mmimgbackup** is **1**. Larger values indicate the display of more detailed information. *n* should be one of the following values:

- 0** Displays only serious errors.
- 1** Displays some information as the command executes, but not for each file. This is the default.
- 2** Displays each chosen file and the scheduled action.
- 3** Displays the same information as 2, plus each candidate file and the applicable rule.
- 4** Displays the same information as 3, plus each explicitly **EXCLUDEd** file and the applicable rule.
- 5** Displays the same information as 4, plus the attributes of candidate and **EXCLUDEd** files.
- 6** Displays the same information as 5, plus non-candidate files and their attributes.

-N {*Node[,Node...] | NodeFile | NodeClass*}

Specifies the nodes that will participate in the backup. This command supports all defined node classes. The default is to run only on the node where the **mmimgbackup** command is running or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

Note: If more than one node is specified, ensure all nodes have the same operating system or unexpected results may occur.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-S *SnapshotName*

Archives the files in the specified global snapshot rather than in the active file system. The snapshot specified cannot be a fileset-level snapshot.

--image *ImageSetName*

Saves the image files using the provided argument as the base set name. Image file names use the following format:

ImageSetName_YYYYMMDD_hh.mm.ss_BBB.sbr

or

ImageSetName_YYYYMMDD_hh.mm.ss.idx

where:

ImageSetName

The default *ImageSetName* is **ImageArchive**.

YYYY

A four-digit year.

MM A two-digit month.

DD A two-digit day.

hh A two-digit hour.

mm A two-digit minute.

ss A two-digit second.

BBB

A three-digit bucket number.

--notsm | **--tsm**

Omits (enables) archiving an image fileset to TSM through the **dsmsc** commands.

--tsm-server *ServerName*

Specifies the server name to provide to the TSM **dsmsc** command used to store the image data files in the TSM server.

POLICY-OPTIONS

The following **mmapplypolicy** options may also be used with **mmimgbackup**:

-A *IscanBuckets*

Specifies the number of buckets of inode numbers (number of inode/filelists) to be created and processed by the parallel inode scan. The default is 17. A bucket will typically represent 1,000,000 in-use inodes.

-a *IscanThreads*

Specifies the number of threads and sort pipelines each node will run during parallel inode scan and policy evaluation. The default is 4.

-D *yyyy-mm-dd[@hh:mm[:ss]]*

Specifies the date and (UTC) time to be used by the **mmimgbackup** command when evaluating the policy rules. The default is the current date and time. If only a date is specified, the time will default to 00:00:00.

-M *name=value*

Indicates a user defined macro specification. There can be more than one **-M** argument. These macro specifications are passed on to the **m4** preprocessor as **-D** specifications.

mmimgbackup

-n *DirThreadLevel*

Specifies the number of threads that will be created and dispatched within each **mmimgbackup** process during the directory scan phase. The default is 24.

-s *LocalWorkDirectory*

Specifies the directory to be used for local temporary storage during command processing. The default directory is **/tmp**.

--single-instance

Ensures that only one instance of **mmimgbackup** is running for each file system at a time.

--sort-buffer-size *Size*

Specifies the size for the main memory buffer to be used by sort command.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmimgbackup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To create a backup image with an *ImageSetName* of **sobar.cluster.fs9**, for the snapshot **snap1** of file system **fs9** where the image is stored in the file system **/backup_images** on the TSM server, issue:

```
mmimgbackup fs9 -S snap1 -g /backup_images --image sobar.cluster.fs9
```

To show that the images are stored on the TSM server, issue this command:

```
dsmls /backup_images
```

The system displays information similar to:

```
/backup_images/4063536/mmPolicy.4260220.51A8A6BF:  
  1088      1088      0  r  sobar.cluster.fs9_20121129_16.19.55.idx  
  4520      4520      0  r  sobar.cluster.fs9_20121129_16.19.55_000.sbr
```

See also

- “mmapplypolicy command” on page 266
- “mmbackupconfig command” on page 290
- “mmimgrestore command” on page 511
- “mmrestoreconfig command” on page 631

Location

```
/usr/lpp/mmfs/bin
```

mmimgrestore command

Restores a single GPFS file system from a metadata image.

Synopsis

```
mmimgrestore Device ImagePath [-g GlobalWorkDirectory]
                    [-L n] [-N {Node[,Node...] | NodeFile | NodeClass}]
                    [--image ImageSetName] [POLICY-OPTIONS]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmimgrestore** command restores a single GPFS file system from a metadata image.

The **mmrestoreconfig** command must be run prior to running the **mmimgrestore** command. For more information, see “Scale Out Backup and Restore (SOBAR)” on page 53 in *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

Device

The device name of the file system whose metadata image is to be restored. The file system must be empty and mounted read-only. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

ImagePath

The fully-qualified path name to an image fileset containing GPFS backup images. The path must be accessible by every node participating in the restore.

-g *GlobalWorkDirectory*

The directory to be used for temporary files that need to be shared between the **mmimgrestore** worker nodes. If not specified, the default working directory will be the *ImagePath* specified.

-L *n*

Controls the level of information displayed by the **mmimgrestore** command. The default for **mmimgrestore** is **1**. Larger values indicate the display of more detailed information. *n* should be one of the following values:

- 0** Displays only serious errors.
- 1** Displays some information as the command executes, but not for each file. This is the default.
- 2** Displays each chosen file and the scheduled action.
- 3** Displays the same information as 2, plus each candidate file and the applicable rule.
- 4** Displays the same information as 3, plus each explicitly **EXCLUDEd** file and the applicable rule.
- 5** Displays the same information as 4, plus the attributes of candidate and **EXCLUDEd** files.
- 6** Displays the same information as 5, plus non-candidate files and their attributes.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the nodes that will participate in the restore. This command supports all defined node classes. The default is to run only on the node where the **mmimgrestore** command is running or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

mmimgrestore

Note: If more than one node is specified, ensure all nodes have the same operating system or unexpected results may occur.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--image *ImageSetName*

Specifies the image set name representing the metadata image to be restored. The *ImageSetName* must match the *ImageSetName_YYYYMMDD_hh.mm.ss* that was created during the backup with the **mmimgbackup** command.

POLICY-OPTIONS

The following **mmapplypolicy** options may also be used with **mmimgrestore**:

-m *ThreadLevel*

The number of threads that will be created and dispatched within each image restore process during the policy execution phase of restore. The default is calculated to divide the work of processing all image files being restored evenly among all nodes specified with **-N**. The valid range is 1 to 20.

-s *LocalWorkDirectory*

Specifies the directory to be used for local temporary storage during command processing. The default directory is **/tmp**.

--single-instance

Ensures that only one instance of **mmimgrestore** is running for each file system at a time.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmimgrestore** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To restore file system **fs9** with data stored in the image with an *ImageSetName* of **sobar.cluster.fs9_20121129_16.19.55** and execute the restore only on AIX nodes, issue:

```
mmimgrestore fs9 /backup_images/406*/mmP* --image sobar.cluster.fs9_20121129_16.19.55 -N aixnodes
```

See also

- “**mmapplypolicy** command” on page 266
- “**mmbackupconfig** command” on page 290
- “**mmimgbackup** command” on page 508
- “**mmrestoreconfig** command” on page 631

Location

/usr/lpp/mmfs/bin

mmimportfs command

Imports into the cluster one or more file systems that were created in another GPFS cluster.

Synopsis

```
mmimportfs {Device | all} -i ImportfsFile [-S ChangeSpecFile]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmimportfs** command, in conjunction with the **mmexportfs** command, can be used to move into the current GPFS cluster one or more file systems that were created in another GPFS cluster. The **mmimportfs** command extracts all relevant file system and disk information from the *ExportFilesysData* file specified with the **-i** parameter. This file must have been created by the **mmexportfs** command.

When **all** is specified in place of a file system name, any disks that are not associated with a file system will be imported as well.

If the file systems being imported were created on nodes that do not belong to the current GPFS cluster, the **mmimportfs** command assumes that all disks have been properly moved, and are online and available to the appropriate nodes in the current cluster.

If any node in the cluster, including the node on which you are running the **mmimportfs** command, does not have access to one or more disks, use the **-S** option to assign NSD servers to those disks.

The **mmimportfs** command attempts to preserve any NSD server assignments that were in effect when the file system was exported.

After the **mmimportfs** command completes, use **mmnsd** to display the NSD server names that are assigned to each of the disks in the imported file system. Use **mmchnsd** to change the current NSD server assignments as needed.

After the **mmimportfs** command completes, use **mmdisk** to display the failure groups to which each disk belongs. Use **mmchdisk** to make adjustments if necessary.

If you are importing file systems into a cluster that already contains GPFS file systems, it is possible to encounter name conflicts. You must resolve such conflicts before the **mmimportfs** command can succeed. You can use the **mmchfs** command to change the device name and mount point of an existing file system. If there are disk name conflicts, use the **mmcrnsd** command to define new disks and specify unique names (rather than let the command generate names). Then replace the conflicting disks using **mmrpldisk** and remove them from the cluster using **mmdelnsd**.

Results

Upon successful completion of the **mmimportfs** command, all configuration information pertaining to the file systems being imported is added to configuration data of the current GPFS cluster.

Parameters

Device | **all**

The device name of the file system to be imported. File system names need not be fully-qualified. *fs0* is as acceptable as */dev/fs0*. Specify **all** to import all GPFS file systems, as well as all disks that do not currently belong to a file system.

mmimportfs

If the specified file system device is a GPFS Native RAID-based file system, then all affected GPFS Native RAID objects will be imported as well. This includes recovery groups, declustered arrays, vdisks, and any other file systems that are based on these objects. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

This must be the first parameter.

-i *ImportfsFile*

The path name of the file containing the file system information. This file must have previously been created with the **mmexportfs** command.

-S *ChangeSpecFile*

The path name of an optional file containing disk stanzas or recovery group stanzas, or both, specifying the changes that are to be made to the file systems during the import step.

Prior to GPFS 3.5, the disk information was specified in the form of disk descriptors defined as:

```
DiskName:ServerList:
```

For backward compatibility, the **mmimportfs** command will still accept the traditional disk descriptors, but their use is discouraged.

Disk stanzas have the following format:

```
%nsd:  
  nsd=NsdName  
  servers=ServerList  
  usage=DiskUsage  
  failureGroup=FailureGroup  
  pool=StoragePool  
  device=DiskName
```

where:

nsd=*DiskName*

Is the name of a disk from the file system being imported. This clause is mandatory for the **mmimportfs** command.

servers=*ServerList*

Is a comma-separated list of NSD server nodes. You can specify up to eight NSD servers in this list. The defined NSD will preferentially use the first server on the list. If the first server is not available, the NSD will use the next available server on the list.

When specifying server nodes for your NSDs, the output of the **mmlscluster** command lists the host name and IP address combinations recognized by GPFS. The utilization of aliased host names not listed in the **mmlscluster** command output may produce undesired results.

If you do not define a *ServerList*, GPFS assumes that the disk is SAN-attached to all nodes in the cluster. If all nodes in the cluster do not have access to the disk, or if the file system to which the disk belongs is to be accessed by other GPFS clusters, you must specify a *ServerList*.

To remove the NSD server list, do not specify a value for *ServerList* (remove or comment out the **servers**=*ServerList* clause of the NSD stanza).

usage=*DiskUsage*

Specifies the type of data to be stored on the disk. If this clause is specified, the value must match the type of usage already in effect for the disk; **mmimportfs** cannot be used to change this value.

failureGroup=*FailureGroup*

Identifies the failure group to which the disk belongs. If this clause is specified, the value must match the failure group already in effect for the disk; **mmimportfs** cannot be used to change this value.

pool=StoragePool

Specifies the storage pool to which the disk is to be assigned. If this clause is specified, the value must match the storage pool already in effect for the disk; **mmimportfs** cannot be used to change this value.

device=DiskName

The block device name of the underlying disk device. This clause is ignored by the **mmimportfs** command.

Recovery group stanzas have the following format:

```
%rg: rgName=RecoveryGroupName
      servers=Primary[,Backup]
```

where:

RecoveryGroupName

Specifies the name of the recovery group being imported.

Primary[,Backup]

Specifies the primary server and, optionally, a backup server to be associated with the recovery group.

Notes:

1. You cannot change the name of a disk. You cannot change the disk usage or failure group assignment with the **mmimportfs** command. Use the **mmchdisk** command for this purpose.
2. All disks that do not have stanzas in *ChangeSpecFile* are assigned the NSD servers that they had at the time the file system was exported. All disks with NSD servers that are not valid are assumed to be SAN-attached to all nodes in the cluster. Use the **mmchnsd** command to assign new or change existing NSD server nodes.
3. Use the **mmchrecoverygroup** command to activate recovery groups that do not have stanzas in *ChangeSpecFile*. The **mmchrecoverygroup** command is documented in *IBM Spectrum Scale RAID: Administration*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmimportfs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To import all file systems in the current cluster, issue this command:

```
mmimportfs all -i /u/admin/exportfile
```

The output is similar to this:

```
mmimportfs: Processing file system fs1 ...
mmimportfs: Processing disk gpfs2nsd
mmimportfs: Processing disk gpfs3nsd
mmimportfs: Processing disk gpfs4nsd
```

mmimportfs

```
mmimportfs: Processing file system fs2 ...
mmimportfs: Processing disk gpfs1nsd1
mmimportfs: Processing disk gpfs5nsd

mmimportfs: Processing disks that do not belong to any file system ...
mmimportfs: Processing disk gpfs6nsd
mmimportfs: Processing disk gpfs1001nsd

mmimportfs: Committing the changes ...

mmimportfs: The following file systems were successfully imported:
    fs1
    fs2
mmimportfs: 6027-1371 Propagating the cluster configuration data to all
    affected nodes. This is an asynchronous process.
```

See also

- “mmexportfs command” on page 485

Location

/usr/lpp/mmfs/bin

mmlinkfileset command

Creates a junction that references the root directory of a GPFS fileset.

Synopsis

```
mmlinkfileset Device FilesetName [-J JunctionPath]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmlinkfileset** command creates a junction at *JunctionPath* that references the root directory of *FilesetName*. The junction is a special directory entry, much like a POSIX hard link, that connects a name in a directory of one fileset, the parent, to the root directory of a child fileset. From the user's viewpoint, a junction always appears as if it were a directory, but the user is not allowed to issue the **unlink** or **rmdir** commands on a junction. Instead, the **mmunlinkfileset** command must be used to remove a junction.

If *JunctionPath* is not specified, the junction is created in the current directory with the name *FilesetName*. The user may use the **mv** command on the directory to move to a new location in the parent fileset, but the **mv** command is not allowed to move the junction to a different fileset.

For information on GPFS filesets, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system that contains the fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

FilesetName

Specifies the name of the fileset to be linked. It must not already be linked into the namespace.

There are no restrictions on linking independent filesets, but a dependent fileset can only be linked inside its own inode space.

-J *JunctionPath*

Specifies the name of the junction. The name must not refer to an existing file system object.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmlinkfileset** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmlinkfileset

Examples

This command links fileset **fset1** in file system **gpfs1** to junction path **/gpfs1/fset1**:

```
mmlinkfileset gpfs1 fset1 -J /gpfs1/fset1
```

The system displays output similar to:

```
Fileset 'fset1' linked at '/gpfs1/fset1'.
```

To confirm the change, issue this command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':  
Name           Status    Path  
root           Linked   /gpfs1  
fset1         Linked   /gpfs1/fset1
```

See also

- “mmchfileset command” on page 365
- “mmcrfileset command” on page 408
- “mmdelfileset command” on page 455
- “mmlsfileset command” on page 532
- “mmunlinkfileset command” on page 687

Location

```
/usr/lpp/mmfs/bin
```

mmlsattr command

Queries file attributes.

Synopsis

```
mmlsattr [-L] [-l]
          [-d | --dump-attr]
          [-n AttributeName | --get-attr AttributeName]
          [-X | --hex-attr] [--hex-attr-name]
          Filename [Filename...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsattr** command to display attributes of a file.

Results

For the specified file, the **mmlsattr** command lists:

- The current number of copies of data for a file and the maximum value
- The number of copies of the metadata for a file and the maximum value
- Whether the Direct I/O caching policy is in effect for a file

Parameters

Filename

The name of the file to be queried. You must enter at least one file name; if you specify more than one, delimit each file name by a space. Wildcard characters are supported in file names; for example, **project*.sched**.

- l Specifies that this command works only with regular files and directories and does not follow symlinks. The default is to follow symlinks.
- L Displays additional file attributes:
 - The assigned storage pool name of the file.
 - The name of the fileset that includes the file.
 - If a file is a snapshot file, the name of the snapshot that includes the file is shown. If the file is a regular file, an empty string is displayed.
 - Whether the file is exposed, ill replicated, ill placed, or unbalanced (displayed under the **flags** heading).
 - Whether the file is immutable.
 - Whether the file is in **appendOnly** mode.
 - The creation time of the file.

-L may be combined with **-d | --dump-attr** to display all extended attribute names and values for each file.

-d | --dump-attr

Displays the names of all extended attributes for each file.

-n *AttributeName* | --get-attr *AttributeName*

Displays the name and value of the specified extended attribute for each file.

mmlsattr

-X | --hex-attr

Displays the attribute value in hex.

--hex-attr-name

Displays the attribute name in hex.

Exit status

0 Successful completion.

nonzero

A failure has occurred. The return code equals the number of files from which the command was not able to get attribute information.

Security

You must have read access to run the **mmlsattr** command.

You may issue the **mmlsattr** command only from a node in the GPFS cluster where the file system is mounted.

Examples

1. To list the attributes of a file, issue this command:

```
mmlsattr -L newfile
```

The system displays information similar to:

```
file name:          newfile
metadata replication: 1 max 2
data replication:   1 max 2
immutable:         no
appendOnly:       no
flags:            directio
storage pool name: system
fileset name:     root
snapshot name:
creation Time:    Wed Feb 22 15:16:29 2012
Misc attributes:  ARCHIVE
```

2. To show the attributes for all files in the root directory of file system **fs0**, issue this command:

```
mmlsattr /fs0/*
```

The system displays information similar to:

```
      replication factors
metadata(max) data(max) file   [flags]
-----
      1 ( 1)  1 ( 1) /fs0/project4.sched
      1 ( 1)  1 ( 1) /fs0/project4.hist
      1 ( 1)  1 ( 1) /fs0/project5.plan
```

3. To show all extended attribute names and values for the file **/ba1/newfile**, issue this command:

```
mmlsattr -d -L /ba1/newfile
```

The system displays information similar to:

```
file name:          /ba1/newfile
metadata replication: 1 max 2
data replication:   1 max 2
immutable:         no
appendOnly:       no
flags:            directio
storage pool name: system
fileset name:     root
snapshot name:
```

```
creation time:      Wed Feb 22 15:16:29 2012
Misc attributes:   ARCHIVE
user.attr1:       "value1"
user.attr:        "val1"
gpfs.DIRECTIO:    "1"
user.eal:         "value1"
```

See also

- “mmchattr command” on page 321

Location

/usr/lpp/mmfs/bin

mmlscallback command

Lists callbacks that are currently registered in the GPFS system.

Synopsis

```
mmlscallback [CallbackIdentifier[,CallbackIdentifier...]] | user | system | all
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlscallback** command to list some or all of the callbacks that are currently registered in the GPFS system.

Parameters

CallbackIdentifier

Indicates the callback for which information is displayed.

user

Indicates all user-defined callbacks. This is the default.

system

Indicates all system-defined callbacks.

all

Indicates all callbacks currently registered with the system.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmlscallback** command

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To list all of the callbacks that are currently in the GPFS system, issue this command:

```
mmlscallback
```

The system displays information similar to:

```
test1
  command      = /tmp/myScript
  event        = startup

test2
  command      = /tmp/myScript2
  event        = shutdown
  parms        = %upNodes
```


To list a specific callback (for example, **test2**) that is currently in the GPFS system, issue this command:

```
mmlscallback test2
```

The system displays information similar to:

```
test2
  command      = /tmp/myScript2
  event        = shutdown
  parms        = %upNodes
```

See also

- “`mmaddcallback` command” on page 226
- “`mmdelcallback` command” on page 448

Location

```
/usr/lpp/mmfs/bin
```

mmlscluster

mmlscluster command

Displays the current configuration information for a GPFS cluster.

Synopsis

```
mmlscluster [--cnfs] [--ces]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlscluster** command to display the current configuration information for a GPFS cluster.

For the GPFS cluster, the **mmlscluster** command displays:

- The cluster name
- The cluster ID
- GPFS UID domain
- The remote shell command being used
- The remote file copy command being used
- The repository type (CCR or server-based)
- The primary GPFS cluster configuration server (if server-based repository)
- The secondary GPFS cluster configuration server (if server-based repository)
- A list of nodes belonging the GPFS cluster

For each node, the command displays:

- The node number assigned to the node by GPFS
- GPFS daemon node interface name
- Primary network IP address
- GPFS administration node interface name
- Designation, such as whether the node is any of the following:
 - quorum node
 - manager node
 - snmp_collector node
 - gateway node

Parameters

--cnfs

Displays information about clustered NFS.

--ces

Displays information about protocol nodes.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmlscluster** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

- To display the current configuration information for the GPFS cluster, issue this command:

```
mmlscluster
```

The system displays information similar to:

```
GPFS cluster information
```

```
=====
```

```
GPFS cluster name:      cluster1.kgn.ibm.com
GPFS cluster id:       680681562214606028
GPFS UID domain:      cluster1.kgn.ibm.com
Remote shell command:  /usr/bin/ssh
Remote file copy command: /usr/bin/scp
Repository type:      CCR
```

Node	Daemon node name	IP address	Admin node name	Designation
1	k164n04.kgn.ibm.com	89.116.68.68	k164n04.kgn.ibm.com	quorum
2	k164n05.kgn.ibm.com	89.116.68.69	k164n05.kgn.ibm.com	quorum
3	k164n06.kgn.ibm.com	89.116.68.70	k164sn06.kgn.ibm.com	quorum-manager

See also

- “mmaddnode command” on page 245
- “mmchcluster command” on page 327
- “mmcrcluster command” on page 403
- “mmdelnode command” on page 460

Location

```
/usr/lpp/mmfs/bin
```

mmlsconfig command

Displays the current configuration data for a GPFS cluster.

Synopsis

```
mmlsconfig [Attribute[,Attribute...]]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsconfig** command to display the requested configuration attributes for a GPFS cluster. If no specific attributes are requested, the command displays all values that were set explicitly by the user. Depending on your configuration, additional information that is set by GPFS might be displayed. If a configuration attribute is not shown in the output of this command, the default value for that attribute, as documented in the **mmchconfig** command, is in effect.

Parameters

Attribute

Specifies the name of the attribute to be displayed. If an attribute has unique values that apply to only a subset of the nodes, the values are followed by the list of affected node names. You can specify more than one attribute in a comma-separated list.

Note: See the **mmchconfig** command for a list of supported attributes.

Exit status

0 Successful completion.

nonzero

A failure occurred.

Security

You must have root authority to run the **mmlsconfig** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster. It must be able to do so without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To display the current configuration data for the GPFS cluster that you are running on, issue this command:

```
mmlsconfig
```

The system displays information similar to the following example:

```
Configuration data for cluster small.cluster:
```

```
-----
```

```
myNodeConfigNumber 1
clusterName small.cluster
clusterId 6339012640885012929
autoload yes
minReleaseLevel 4.1.0.0
dmapiFileHandleSize 32
```

```
[c6f1c3vp3]
pagepool 512M
[common]
adminMode central
```

File systems in cluster small.cluster:

```
-----
/dev/fs1
/dev/gpfs1
```

2. To display the current values for the **maxblocksize** and **pagepool** attributes, issue this command:

```
mmlsconfig maxblocksize,pagepool
```

The system displays information similar to the following example:

```
maxblocksize 1M
pagepool 1G
pagepool 512M [c6f1c3vp3]
```

3. To display the current value for the **cipherList** attribute, issue this command:

```
mmlsconfig cipherList
```

The system displays information similar to the following example:

```
cipherList AUTHONLY
```

See also

- “mmchcluster command” on page 327
- “mmchconfig command” on page 331
- “mmcrcluster command” on page 403

Location

```
/usr/lpp/mmfs/bin
```

mmlsdisk command

Displays the current configuration and state of the disks in a file system.

Synopsis

```
mmlsdisk Device [-d "DiskName[;DiskName...]" [-e] [-L]
```

or

```
mmlsdisk Device [-d "DiskName[;DiskName...]" {-m | -M}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsdisk** command to display the current state of the disks in the file system.

The **mmlsdisk** command may be run against a mounted or unmounted file system.

For each disk in the list, the **mmlsdisk** command displays the following:

- disk name
- driver type
- logical sector size (under the heading “sector size”)
- failure group
- whether it holds metadata
- whether it holds data
- status:
 - ready** Normal status.
 - suspended**
or
to be emptied Indicates that data is to be migrated off this disk.
 - being emptied** Transitional status in effect while a disk deletion is pending.
 - emptied** Indicates that data is already migrated off this disk.
 - replacing** Transitional status in effect for old disk while replacement is pending.
 - replacement** Transitional status in effect for new disk while replacement is pending.
- availability:
 - up** The disk is available to GPFS for normal **read** and **write** operations.
 - down** No **read** and **write** operations can be performed on this disk.
 - recovering** An intermediate state for disks coming up, during which GPFS verifies and corrects data. **write** operations can be performed while a disk is in this state, but **read** operations cannot (because data on the disk being recovered might be stale until the **mmchdisk start** command completes).

unrecovered

The disk was not successfully brought up.

- disk ID
- storage pool to which the disk is assigned

Parameters*Device*

The device name of the file system to which the disks belong. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

-d "DiskName[;DiskName...]"

The name of the disks for which you want to display current configuration and state information.

When you enter multiple values for *DiskName*, separate them with semicolons and enclose the list in quotation marks.

```
"gpfs3nsd;gpfs4nsd;gpfs5nsd"
```

Options

- e** Display all of the disks in the file system that do not have an availability of **up** and a status of **ready**. If all disks in the file system are **up** and **ready**, the message displayed is:
6027-623 All disks up and ready
- L** Displays an extended list of the disk parameters, including the disk ID field and the **remarks** field. The **remarks** column shows the current file system descriptor quorum assignments, and displays the excluded disks. The **remarks** field contains **desc** for all disks assigned as the file system descriptor holders and **excl** for all excluded disks.
- M** Displays whether I/O requests to the disk are satisfied on the local node, or using an NSD server. If the I/O is done using an NSD server, shows the NSD server name and the underlying disk name on that server node.
- m** Displays whether I/O requests to the disk are satisfied on the local node, or using an NSD server. The scope of this option is the node on which the **mmlsdisk** command is issued.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

As root, the command can also do an **mmlsdisk** on remote file systems.

If you are a non-root user, you may specify only file systems that belong to the same cluster as the node on which the **mmlsdisk** command was issued.

The **mmlsdisk** command does not work if GPFS is down.

mmlsdisk

Examples

1. To display the current state of **gpfs2nsd**, issue this command:

```
mmlsdisk /dev/fs0 -d gpfs2nsd
```

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	storage pool
gpfs2nsd	nsd	512	4002	yes	yes	ready	up	system

Note: In this output, “sector size” refers to logical sector size.

2. To display the current states of **gpfs2nsd**, **gpfs3nsd**, and **gpfs4nsd**, and display their respective disk ids and the descriptor quorum assignment, issue this command:

```
mmlsdisk /dev/fs0 -d "gpfs2nsd;gpfs3nsd;gpfs4nsd" -L
```

The system displays information similar to:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	disk id	storage pool	remarks
gpfs2nsd	nsd	512	4002	yes	yes	ready	up	2	system	desc
gpfs3nsd	nsd	512	4002	yes	yes	ready	up	3	system	
gpfs4nsd	nsd	512	4002	yes	yes	ready	up	4	system	

Number of quorum disks: 3
Read quorum value: 2
Write quorum value: 2

Note: In this output, “sector size” refers to logical sector size.

3. After the `mmchdisk fs0 empty -d gpfs1nsd` command has been issued, you can view the current state of **gpfs1nsd** by issuing the following command:

```
mmlsdisk fs0 -L
```

In IBM Spectrum Scale V4.1.1 and later, the system displays information similar to the following example:

disk name	driver type	sector size	failure group	holds metadata	holds data	status	availability	disk id	storage pool
gpfs1nsd	nsd	512	-1	Yes	Yes	to be emptied	up	1	system
gpfs2nsd	nsd	512	-1	Yes	Yes	to be emptied	up	2	system
gpfs3nsd	nsd	512	-1	Yes	Yes	ready	up	3	system
gpfs4nsd	nsd	512	-1	Yes	Yes	ready	up	4	system

Number of quorum disks: 3
Read quorum value: 2
Write quorum value: 2

Attention: Due to an earlier configuration change the file system may contain data that is at risk of being lost.

4. To display whether the I/O is performed locally or using an NSD server, the NSD server name, and the underlying disk name for the file system named **test**, issue this command:

```
mmlsdisk test -M
```

The system displays information similar to:

Disk name	I/O performed on node	Device	Availability
gpfs7nsd	localhost	/dev/hdisk12	up
gpfs10nsd	k5n88.kgn.ibm.com	/dev/hdisk13	up
gpfs4nsd	localhost	/dev/hdisk10	up

5. To display the same information as in the previous example, but limited to the node on which the command is issued, issue this command:

```
mmlsdisk test -m
```


The system displays information similar to:

Disk name	I/O performed on node	Device	Availability
gpfs7nsd	localhost	/dev/hdisk12	up
gpfs10nsd	k5n88.kgn.ibm.com	-	up
gpfs4nsd	localhost	/dev/hdisk10	up

See also

- “mmaddisk command” on page 239
- “mmchdisk command” on page 354
- “mmdeldisk command” on page 449
- “mmrpldisk command” on page 648

Location

/usr/lpp/mmfs/bin

mmlsfileset command

Displays attributes and status for GPFS filesets.

Synopsis

```
mmlsfileset Device
  [[Fileset[,Fileset...]] [-J Junction[,Junction...]] | -F FileName]
  [-d [--block-size {BlockSize | auto}]] [-i] [-L] [-X] [--afm]
  [--deleted] [--iam-mode]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmlsfileset** command to display information for the filesets that belong to a given GPFS file system. The default is to display information for all filesets in the file system. You may choose to display information for only a subset of the filesets.

The operation of the **-L** flag omits the attributes listed without it, namely status and junction path. In addition, if the fileset has status Deleted, then **-L** also displays the name of the latest snapshot that includes the fileset in place of the root inode number and parent fileset identifier.

The attributes displayed are:

- Name of the fileset
- Status of the fileset (when the **-L** flag is omitted)
- Junction path to the fileset (when the **-L** flag is omitted)
- Fileset identifier (when the **-L** flag is included)
- Root inode number, if not deleted (when the **-L** flag is included)
- Parent fileset identifier, if not deleted (when the **-L** flag is included)
- Latest including snapshot, if deleted (when the **-L** flag is included)
- Creation time (when the **-L** flag is included)
- Inode space (when the **-L** flag is included)
- Number of inodes in use (when the **-i** flag is included)
- Data size (when the **-d** flag is included)
- Comment (when the **-L** flag is included)
- Caching-related information (when the **--afm** flag is included)
- Value of the permission change flag (when the **-X** flag is used to generate stanza output)
- Integrated archive manger (IAM) mode information

For information on GPFS filesets, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system that contains the fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

Fileset

Specifies a comma-separated list of fileset names.

-J Junction

Specifies a comma-separated list of path names. They are not restricted to fileset junctions, but may name any file or directory within the filesets to be listed.

-F FileName

Specifies the name of a file containing either fileset names or path names. Each line must contain a single entry. All path names must be fully-qualified.

-d Displays the amount of storage in use for the fileset.

This operation requires an amount of time that is proportional to the size of the file system; therefore, it can take several minutes or even hours on a large and heavily-loaded file system.

This optional parameter can impact overall system performance. Avoid running the **mmlsfileset** command with this parameter frequently or during periods of high file system activity.

--block-size {BlockSize | auto}

Specifies the unit in which the number of blocks is displayed. The value must be of the form $[n]K$, $[n]M$, $[n]G$ or $[n]T$, where n is an optional integer in the range 1 to 1023. The default is 1K. If **auto** is specified, the number of blocks is automatically scaled to an easy-to-read value.

-i Displays the number of inodes in use for the fileset.

This operation requires an amount of time that is proportional to the number of inodes in the file system; therefore, it can take several minutes or even hours on a large and heavily-loaded file system.

Information about the number of inodes in the fileset can be retrieved more efficiently with the following command, if quota management has been enabled for the file system:

```
mmrepquota -j FileSystem
```

-L Displays additional information for the fileset. This includes:

- Fileset identifier
- Root inode number
- Parent identifier
- Fileset creation time
- Inode space
- User defined comments, if any

If the fileset is a dependent fileset, **dpnd** will be displayed next to the inode space identifier.

-X Generates stanza output containing the following:

- The same information presented by the **-L** flag
- The value of the permission change flag
- The same information presented by the **--afm** flag

--afm

Displays caching-related information for the fileset.

--deleted

Displays only the filesets with a status of Deleted.

--iam-mode

Displays integrated archive manager (IAM) mode information.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmlsfileset

Security

Fileset owners can run the **mmlsfileset** command with the **-L**, **-d**, and **-i** options.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This command displays fileset information for all filesets in file system **gpfs1**:

```
mmlsfileset gpfs1
```

The system displays information similar to:

Filesets in file system 'gpfs1':

Name	Status	Path
root	Linked	/gpfs1
fset1	Linked	/gpfs1/fset1
fset2	Linked	/gpfs1/fset1/fset2

2. These commands display information for a file system with filesets and snapshots. Note that deleted filesets that are saved in snapshots are displayed with the name enclosed in parentheses.

- a. Command:

```
mmlsfileset fs1 -d -i
```

The system displays information similar to:

Filesets in file system 'fs1':

Name	Status	Path	Inodes	Data (in KB)
root	Linked	/gpfs	3	53528
(gone)	Deleted	/gpfs/.snapshots/Snap17/gone	0	0
TestF4	Linked	/gpfs/test-f4	3	24
TestF3	Linked	/gpfs/test-f4/dir1/f3	2	16
TestF2	Unlinked	--	98	784
TestF5	Linked	<TestF2>/subdir/f5	1	8

- b. Command:

```
mmlsfileset fs1 --deleted
```

The system displays information similar to:

Filesets in file system 'fs1':

Name	Status	Path
(gone)	Deleted	/gpfs/.snapshots/Snap17/gone

- c. Command:

```
mmlsfileset fs1 --afm
```

The system displays information similar to:

Filesets in file system 'fs1':

Name	Status	Path	afmTarget
root	Linked	/gpfs/fs1	--
ro1	Linked	/gpfs/fs1/ro1	hs21n45:/gpfs/fs1/ro1
sw1	Linked	/gpfs/fs1/sw1	hs21n45:/gpfs/fs1/sw1
lu1	Linked	/gpfs/fs1/lu1	hs21n45:/gpfs/fs1/lu1

- d. Command:

```
mmlsfileset fs1 -L
```

The system displays information similar to:

Filesets in file system 'fs1':

Name	Id	RootInode	ParentId	Created	InodeSpace	MaxInodes	AllocInodes	Comment
root	0	3	--	Mon Jan 23 18:59:36 2012	0	1000064	65792	root fileset
TestF4	2	59446	0	Wed Feb 1 09:28:50 2012	0	0	0	4th in series
TestF3	3	59435	2	Wed Feb 1 09:28:52 2012	0	0	0	
(gone)	1	latest:	Snap17	Wed Feb 1 09:28:46 2012	0	0	0	Not forgotten

```

TestF2      4      59437    -- Wed Feb  1 09:28:52 2012    0          0          0
TestF5      5      7017      4 Wed Feb  1 09:28:53 2012    0          0          0 Number 5
FsetF1-V2   6      131075    -- Wed Feb  1 09:28:55 2012    1      100096    100096
FsetF2-V2   7      262147    -- Wed Feb  1 09:28:56 2012    2      100096    100096
FsetF2-V2-lite 9      263680    -- Wed Feb  1 09:28:59 2012    2 dpnd    0          0
FsetF3-V2   8      393219    -- Wed Feb  1 09:28:57 2012    3      100096    100096

```

e. Command:

```
mmlsfileset fs1 sw1 --afm -L
```

The system displays information similar to:

Filesets in file system 'fs1':

Attributes for fileset sw:

```

=====
Status                Linked
Path                  /gpfs/fs1/sw
afm-associated        Yes
Target                c2m3n06:/gpfs/fs2
Mode                  single-writer
File Lookup Refresh Interval 30 (default)
File Open Refresh Interval 30 (default)
Dir Lookup Refresh Interval 60 (default)
Dir Open Refresh Interval 60 (default)
Async Delay           15 (default)
Expiration Timeout    disable
Recovery Point Objective  disable (default)
Last pSnapId          0
Display Home Snapshots no

```

f. Command:

```
mmlsfileset fs1 TestF2,TestF5 -J /gpfs/test-f4/dir1,/gpfs/test-f4/dir1/f3/dir2/
```

The system displays information similar to:

Filesets in file system 'fs1':

```

Name      Status   Path
TestF2    Unlinked --
TestF5    Linked   <TestF2>/subdir/f5
TestF4    Linked   /gpfs/test-f4
TestF3    Linked   /gpfs/test-f4/dir1/f3

```

g. Command:

```
mmlsfileset gpfsha --deleted
```

The system displays information similar to:

Filesets in file system 'gpfsha':

```

Name      Status   Path
(fset17) Deleted /gpfsha/.snapshots/snap20/fset17

```

See also

- “mmchfileset command” on page 365
- “mmcrfileset command” on page 408
- “mmdelfileset command” on page 455
- “mmlinkfileset command” on page 517
- “mmunlinkfileset command” on page 687

Location

/usr/lpp/mmfs/bin

mmlsfs command

Displays file system attributes.

Synopsis

```
mmlsfs {Device | all | all_local | all_remote} [-A] [-B] [-d] [-D]
        [-E] [-f] [-i] [-I] [-j] [-k] [-K] [-L] [-m] [-M] [-n] [-o]
        [-P] [-Q] [-r] [-R] [-S] [-t] [-T] [-V] [-z]
        [--create-time] [--encryption] [--fastea] [--filesetdf]
        [--inode-limit] [--is4KAligned] [--log-replicas] [--mount-priority]
        [--perfileset-quota] [--rapid-repair] [--write-cache-threshold]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsfs** command to list the attributes of a file system.

Depending on your configuration, additional information that is set by GPFS may be displayed to assist in problem determination when contacting the IBM Support Center.

Results

If you do not specify any options, all attributes of the file system are displayed. When you specify options, only those attributes specified are listed, in the order issued in the command. Some parameters are preset for optimum performance and, although they display in the **mmlsfs** command output, you cannot change them.

Parameters

The following parameter must be the first parameter:

Device | **all** | **all_local** | **all_remote**

Device

Indicates the device name of the file system for which information is displayed. File system names do not need to be fully qualified. *fs0* is as acceptable as */dev/fs0*.

all

Indicates all file systems that are known to this cluster.

all_local

Indicates all file systems that are owned by this cluster.

all_remote

Indicates all file systems that are owned by another cluster.

This must be the first parameter.

The following optional parameters, when used, must be provided after the *Device* | **all** | **all_local** | **all_remote** parameter:

- A Displays if and when the file system is automatically mounted.
- B Displays the size of the data block, in bytes.
- d Displays the names of all of the disks in the file system.
- D Displays the type of file locking semantics that are in effect (**nfs4** or **posix**).

- E Displays the exact **mtime** values reported.
- f Displays the minimum fragment size, in bytes.
- i Displays the inode size, in bytes.
- I Displays the indirect block size, in bytes.
- j Displays the block allocation type.
- k Displays the type of authorization supported by the file system.
- K Displays the strict replication enforcement.
- L Displays the internal log file size.
- m Displays the default number of metadata replicas.
- M Displays the maximum number of metadata replicas.
- n Displays the estimated number of nodes for mounting the file system.
- o Displays the additional mount options.
- P Displays the storage pools defined within the file system.
- Q Displays which quotas are currently enforced on the file system.
- r Displays the default number of data replicas.
- R Displays the maximum number of data replicas.
- S Displays whether the updating of **atime** is suppressed for the **gpfs_stat()**, **gpfs_fstat()**, **stat()**, and **fstat()** calls.
- t Displays the Windows drive letter.
- T Displays the default mount point.
- V Displays the current format version of the file system.
- z Displays whether DMAPAPI is enabled for this file system.
- create-time**
Displays the creation time of the file system.
- encryption**
Displays a **yes** or **no** value indicating whether encryption is enabled. This value cannot be changed with the **mmchfs** command. When the cluster is created this value is set to **no**. When an encryption policy is established for the file system, the value is set to **yes**.
- fastea**
Displays a **yes** or **no** value indicating whether fast external attributes is enabled. Displays a **migrating** value if migration was initiated with **mmmigratefs --fastea** but is not yet complete.
- filesetdf**
Displays a **yes** or **no** value indicating whether **filesetdf** is enabled; if **yes**, the **mmdf** command reports numbers based on the quotas for the fileset and not for the total file system.
- inode-limit**
Displays the maximum number of files in the file system.
- is4KAligned**
Displays whether file systems are formatted to be 4K aligned.
- log-replicas**
Displays the number of recovery log replicas. If a value of **0** is displayed, the number of recovery log replicas is the same as the number of metadata replicas currently in effect for the file system.

mmlsfs

--mount-priority

Displays the assigned mount priority.

--perfileset-quota

Displays the per-fileset quota.

--rapid-repair

Displays a **yes** or **no** value indicating whether the per-block replication tracking and repair feature is enabled.

--write-cache-threshold

Displays the threshold below which synchronous writes will be initially buffered in the highly-available write cache before being written back to primary storage.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

As root, a user can also issue the **mmlsfs** on remote file systems.

If you are a non-root user, you may specify only file systems that belong to the same cluster as the node on which the **mmlsfs** command was issued.

Examples

If you issue the **mmlsfs** command with no options for the file system **gpfs1**:

```
mmlsfs gpfs1
```

The system displays information similar to this:

flag	value	description
-f	8192	Minimum fragment size in bytes
-i	4096	Inode size in bytes
-I	16384	Indirect block size in bytes
-m	2	Default number of metadata replicas
-M	2	Maximum number of metadata replicas
-r	2	Default number of data replicas
-R	2	Maximum number of data replicas
-j	cluster	Block allocation type
-D	nfs4	File locking semantics in effect
-k	all	ACL semantics in effect
-n	32	Estimated number of nodes that will mount file system
-B	262144	Block size
-Q	user;group;fileset	Quotas accounting enabled
	user;group;fileset	Quotas enforced
	none	Default quotas enabled
--perfileset-quota	no	Per-fileset quota enforcement
--filesetdf	no	Fileset df enabled?
-V	14.20 (4.1.1.0)	File system version
--create-time	Fri Jun 12 18:39:47 2015	File system creation time
-z	no	Is DMAPI enabled?
-L	134217728	Logfile size

-E	yes	Exact mtime mount option
-S	no	Suppress atime mount option
-K	whenpossible	Strict replica allocation option
--fastea	yes	Fast external attributes enabled?
--encryption	no	Encryption enabled?
--inode-limit	607488	Maximum number of inodes in all inode spaces
--log-replicas	2	Number of log replicas
--is4KAligned	yes	is4KAligned?
--rapid-repair	yes	rapidRepair enabled?
--write-cache-threshold	65536	HAWC Threshold (max 65536)
-P	system	Disk storage pools in file system
-d	nsd20;nsd21;nsd3	Disks in file system
-A	yes	Automatic mount option
-o	none	Additional mount options
-T	/gpfs1	Default mount point
--mount-priority	0	Mount priority

If you issue the **mmlsfs** command with the **all** option:

```
mmlsfs all -A
```

The system displays information similar to:

```
File system attributes for /dev/fs1:
```

```
=====
```

flag	value	description
-A	yes	Automatic mount option

```
File system attributes for /dev/gpfs1:
```

```
=====
```

flag	value	description
-A	yes	Automatic mount option

See also

- “mmcrfs command” on page 414
- “mmchfs command” on page 371
- “mmdelfs command” on page 458

Location

```
/usr/lpp/mmfs/bin
```

mmlslicense command

Displays information about the GPFS node licensing designation.

Synopsis

`mmlslicense [-L]`

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlslicense** command to display the number of GPFS client, FPO, and server licenses assigned to the nodes in the cluster.

For information on GPFS license designation, see the topic "GPFS license designation" in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Parameters

- L Displays detailed information about the license type associated with each of the nodes in the cluster. An asterisk after the license type indicates insufficient license level for the roles that the node performs.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmlslicense** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To display the summary information about the type and number of GPFS licenses associated with the nodes in the cluster, issue this command:

```
mmlslicense
```

The system displays information similar to:

Summary information

```
-----  
Number of nodes defined in the cluster:          4  
Number of nodes with server license designation: 1  
Number of nodes with client license designation: 2  
Number of nodes still requiring server license designation: 1  
Number of nodes still requiring client license designation: 1
```

To display detailed information about the type of GPFS licenses associated with each of the nodes in the cluster, issue this command:

```
mmlslicense -L
```

The system displays information similar to:

Node name	Required license	Designated license
k145n05.kgn.ibm.com	server	server
k145n06.kgn.ibm.com	server	client *
k145n07.kgn.ibm.com	client	client
k145n08.kgn.ibm.com	client	none *

Summary information

Number of nodes defined in the cluster:	4
Number of nodes with server license designation:	1
Number of nodes with client license designation:	2
Number of nodes still requiring server license designation:	1
Number of nodes still requiring client license designation:	1

See also

- “mmchlicense command” on page 377

Location

```
/usr/lpp/mmfs/bin
```

mmlsmgr command

Displays which node is the file system manager for the specified file systems or which node is the cluster manager.

Synopsis

```
mmlsmgr [Device [Device...]]
```

or

```
mmlsmgr -C RemoteClusterName
```

or

```
mmlsmgr -c
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsmgr** command to display which node is the file system manager or cluster manager for the file system.

If you do not provide a *Device* operand, file system managers for all file systems within the current cluster for which a file system manager has been appointed are displayed.

Parameters

Device

The device names of the file systems for which the file system manager information is displayed.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

If no file system is specified, information about all file systems is displayed.

-C *RemoteClusterName*

Displays the name of the nodes that are file system managers in cluster *RemoteClusterName*.

-c Displays the current cluster manager node.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

As root, a user can also issue the **mmlsmgr** on remote file systems.

If you are a non-root user, you may specify only file systems that belong to the same cluster as the node on which the **mmlsmgr** command was issued.

Examples

1. To display the file system manager node information for all the file systems, issue this command:

```
mmlsmgr
```

The system displays information similar to:

```
file system      manager node
-----
fs3              9.114.94.65 (c154n01)
fs2              9.114.94.73 (c154n09)
fs1              9.114.94.81 (c155n01)
```

Cluster manager node: 9.114.94.65 (c154n01)

The output shows the device name of the file system and the file system manager's node number and name, in parenthesis, as they are recorded in the GPFS cluster data.

2. To display the file system manager information for file systems **gpfs2** and **gpfs3**, issue this command:

```
mmlsmgr gpfs2 gpfs3
```

The system displays information similar to:

```
file system      manager node [from 199.116.68.69 (k156gn02)]
-----
gpfs2           199.116.68.70 (k154gn02)
gpfs3           199.116.68.72 (ko1t2g_r1b42)
```

See also

- “mmchmgr command” on page 379

Location

```
/usr/lpp/mmfs/bin
```

mmlsmount

mmlsmount command

Lists the nodes that have a given GPFS file system mounted.

Synopsis

```
mmlsmount {Device | all | all_local | all_remote | {-F DeviceFileName}} [-L]
          [-C {all | all_remote | ClusterName[,ClusterName...]}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmlsmount** command reports if a file system is in use at the time the command is issued. A file system is considered to be in use if it is explicitly mounted with the **mount** or **mmount** command, or if it is mounted internally for the purposes of running some other GPFS command. For example, when you run the **mmrestripefs** command, the file system will be internally mounted for the duration of the command. If **mmlsmount** is issued in the interim, the file system will be reported as being in use by the **mmlsmount** command but, unless it is explicitly mounted, will not show up in the output of the **mount** or **df** commands.

Parameters

Device | **all** | **all_local** | **all_remote** | {-F *DeviceFileName*}

Indicates the file system or file systems for which information is displayed.

Device

Indicates the device name of the file system for which information is displayed. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

all

Indicates all file systems known to this cluster.

all_local

Indicates all file systems owned by this cluster.

all_remote

Indicates all file systems owned by another cluster.

-F *DeviceFileName*

Specifies a file containing the device names, one per line, of the file systems for which information is displayed.

This must be the first parameter.

Options

-C {**all** | **all_remote** | *ClusterName*[,*ClusterName*...]}

Specifies the clusters for which mount information is requested. If one or more *ClusterName* is specified, only the names of nodes that belong to these clusters and have the file system mounted are displayed. The dot character (".") can be used in place of the cluster name to denote the local cluster.

Option **-C all_remote** denotes all clusters other than the one from which the command was issued.

Option **-C all** refers to all clusters, local and remote, that can have the file system mounted. Option **-C all** is the default.

-L Specifies to list the nodes that have the file system mounted.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

If you are a non-root user, you may specify only file systems that belong to the same cluster as the node on which the **mmlsmount** command was issued.

Examples

1. To see how many nodes have file system **fs2** mounted, issue this command:

```
mmlsmount fs2
```

The system displays output similar to:

```
File system fs2 is mounted on 3 nodes.
```

2. To display all mounted file systems:

```
mmlsmount all
```

The system displays output similar to:

```
File system fs1 is mounted on 17 nodes.
```

```
File system remotefs1 (remote.cluster:fs1) is mounted on 17 nodes.
```

3. To display all remotely mounted file systems:

```
mmlsmount all_remote
```

The system displays output similar to:

```
File system remotefs1 (remote.cluster:fs1) is mounted on 17 nodes.
```

4. To list the nodes having all file systems mounted:

```
mmlsmount all -L
```

The system displays output similar to:

```
File system fs1 is mounted on 3 nodes:
```

```
192.168.105.32 c6f1c3vp2
```

```
192.168.105.31 c6f1c3vp1
```

```
192.168.105.34 c6f1c3vp4
```

```
File system gpfs1 is not mounted.
```

See also

- “mmount command” on page 568
- “mmumount command” on page 684

Location

```
/usr/lpp/mmfs/bin
```

mmlsnodeclass command

Displays node classes defined in the system.

Synopsis

```
mmlsnodeclass [ClassName[,ClassName...]] | --user | --system | --all]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsnodeclass** command to display node classes defined in the system.

Parameters

ClassName

Displays the specified node class.

--user

Displays all user-defined node classes. This is the default.

--system

Displays all system-defined node classes.

--all

Displays both the system-defined and user-defined node classes.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmlsnodeclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To display the current user-defined node classes, issue this command:

```
mmlsnodeclass
```

The system displays information similar to:

Node Class Name	Members
-----------------	---------

```
-----  
siteA                linuxNodes
```

2. To display the system-defined node classes, issue this command:

```
mmlsnodeclass --system
```

The system displays information similar to:

Node Class Name	Members
aixNodes	c8f2c4vp1,c8f2c4vp2
all	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2,c8f2c4vp3
clientLicense	c8f2c4vp3
clientNodes	c8f2c4vp3
cnfsNodes	
disabledCnfsNodes	
enabledCnfsNodes	
linuxNodes	c8f2c1vp4,c8f2c4vp3
managerNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
nonAixNodes	c8f2c1vp4,c8f2c4vp3
nonCnfsNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp3,c8f2c4vp2
nonLinuxNodes	c8f2c4vp1,c8f2c4vp2
nonNsdNodes	c8f2c4vp3
nonQuorumNodes	c8f2c4vp3
nonWindowsNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp3,c8f2c4vp2
nsdNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
quorumNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
serverLicense	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
windowsNodes	

3. To display all node classes defined in the system, issue this command:

```
mmlsnodeclass --all
```

The system displays information similar to:

Node Class Name	Members
aixNodes	c8f2c4vp1,c8f2c4vp2
all	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2,c8f2c4vp3
clientLicense	c8f2c4vp3
clientNodes	c8f2c4vp3
cnfsNodes	
disabledCnfsNodes	
enabledCnfsNodes	
linuxNodes	c8f2c1vp4,c8f2c4vp3
managerNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
nonAixNodes	c8f2c1vp4,c8f2c4vp3
nonCnfsNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp3,c8f2c4vp2
nonLinuxNodes	c8f2c4vp1,c8f2c4vp2
nonNsdNodes	c8f2c4vp3
nonQuorumNodes	c8f2c4vp3
nonWindowsNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp3,c8f2c4vp2
nsdNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
quorumNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
serverLicense	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2
windowsNodes	
siteA	linuxNodes

4. To display only the nodes that are quorum nodes, issue this command:

```
mmlsnodeclass quorumNodes
```

The system displays information similar to:

Node Class Name	Members
quorumNodes	c8f2c1vp4,c8f2c4vp1,c8f2c4vp2

See also

- “mmchnodeclass command” on page 385
- “mmcrnodeclass command” on page 424
- “mmdelnodeclass command” on page 463

mmlsnodeclass

Location

/usr/lpp/mmfs/bin

mmlsnsd command

Displays Network Shared Disk (NSD) information for the GPFS cluster.

Synopsis

```
mmlsnsd [-a | -F | -f Device | -d "DiskName[;DiskName...]"
         [-L | -m | -M | -X] [-v]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlsnsd** command to display the current information for the NSDs belonging to the GPFS cluster. The default is to display information for all NSDs defined to the cluster (**-a**). Otherwise, you may choose to display the information for a particular file system (**-f**) or for all disks that do not belong to any file system (**-F**).

Parameters

-a Display information for all of the NSDs belonging to the GPFS cluster. This is the default.

-f Device

The device name of the file system for which you want NSD information displayed. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

-F Display the NSDs that are not in use.

-d DiskName[;DiskName...]

The name of the NSDs for which you want information displayed. When you enter multiple *DiskNames*, separate them with semicolons and enclose the entire string of disk names in quotation marks:

```
"gpfs3nsd;gpfs4nsd;gpfs5nsd"
```

Options

-L Displays the information in a long format that shows the NSD identifier.

-m Maps the NSD name to its disk device name on the local node and, if applicable, on the NSD server nodes.

-M Maps the NSD names to its disk device name on all nodes.

This is a slow operation and its usage is suggested for problem determination only.

-v Specifies that the output should contain error information, where available.

-X Maps the NSD name to its disk device name on the local node and, if applicable, on the NSD server nodes. The **-X** option also displays extended information for the NSD volume ID and information such as NSD server status and Persistent Reserve (PR) enablement in the Remarks field. Using the **-X** option is a slow operation and is recommended only for problem determination.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

mmlsnsd

Security

You must have root authority to issue the **mmlsnsd** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To display the default information for all of the NSDs belonging to the cluster, issue this command:

```
mmlsnsd
```

The system displays information similar to:

File system	Disk name	NSD servers
fs2	hd3n97	c5n97g,c5n98g,c5n99g
fs2	hd4n97	c5n97g,c5n98g,c5n99g
fs2	hd5n98	c5n98g,c5n97g,c5n99g
fs2	hd6n98	c5n98g,c5n97g,c5n99g
fs2	hd7n97	c5n97g,c5n98g,c5n99g
fs2	hd8n97	c5n97g,c5n98g,c5n99g
fs2	hd9n97	c5n97g,c5n98g,c5n99g
fs2	hd10n98	c5n98g,c5n97g,c5n99g
fs2	hd11n98	c5n98g,c5n97g
fs2	hd12n98	c5n98g,c5n97g
fs2	sdbnsd	c5n94g,c5n96g
fs2	sdcnscd	c5n94g,c5n96g
fs2	sddnsd	c5n94g,c5n96g
fs2	sdensd	c5n94g,c5n96g
fs2	sdgnscd	c5n94g,c5n96g
fs2	sdfnsd	c5n94g,c5n96g
fs2	sdhnsd	c5n94g,c5n96g
(free disk)	hd2n97	c5n97g,c5n98g

2. To display all of the NSDs attached to the node from which the command is issued, issue this command:

```
mmlsnsd -m
```

The system displays information similar to:

Disk name	NSD volume ID	Device	Node name	Remarks
hd10n98	0972846245C8E93C	/dev/hd10n98	c5n97g	server node
hd10n98	0972846245C8E93C	/dev/hd10n98	c5n98g	server node
hd11n98	0972846245C8E93F	/dev/hd11n98	c5n97g	server node
hd11n98	0972846245C8E93F	/dev/hd11n98	c5n98g	server node
hd12n98	0972846245C8E941	/dev/hd12n98	c5n97g	server node
hd12n98	0972846245C8E941	/dev/hd12n98	c5n98g	server node
hd2n97	0972846145C8E924	/dev/hdisk2	c5n97g	server node
hd2n97	0972846145C8E924	/dev/hdisk2	c5n98g	server node
hd3n97	0972846145C8E927	/dev/hdisk3	c5n97g	server node
hd3n97	0972846145C8E927	/dev/hdisk3	c5n98g	server node
hd4n97	0972846145C8E92A	/dev/hdisk4	c5n97g	server node
hd4n97	0972846145C8E92A	/dev/hdisk4	c5n98g	server node
hd5n98	0972846245EB501C	/dev/hdisk5	c5n97g	server node
hd5n98	0972846245EB501C	/dev/hdisk5	c5n98g	server node
hd6n98	0972846245DB3AD8	/dev/hdisk6	c5n97g	server node
hd6n98	0972846245DB3AD8	/dev/hdisk6	c5n98g	server node
hd7n97	0972846145C8E934	/dev/hd7n97	c5n97g	server node

3. To display all of the NSDs in the GPFS cluster in extended format, issue this command:

```
mmlsnsd -L
```

The system displays information similar to:

File system	Disk name	NSD volume ID	NSD servers
fs2	hd3n97	0972846145C8E927	c5n97g,c5n98g
fs2	hd4n97	0972846145C8E92A	c5n97g,c5n98g
fs2	hd5n98	0972846245EB501C	c5n98g,c5n97g
fs2	hd6n98	0972846245DB3AD8	c5n98g,c5n97g
fs2	sdbnsd	0972845E45C8E8ED	c5n94g,c5n96g
fs2	sdcnsd	0972845E45C8E8F6	c5n94g,c5n96g
fs2	sddnsd	0972845E45F83FDB	c5n94g,c5n96g
fs2	sdensd	0972845E45C8E909	c5n94g,c5n96g
fs2	sdgnsd	0972845E45C8E912	c5n94g,c5n96g
fs2	sdfnsd	0972845E45F02E81	c5n94g,c5n96g
fs2	sdhnsd	0972845E45C8E91C	c5n94g,c5n96g
gpfs1	hd2n97	0972846145C8E924	c5n97g,c5n98g

4. To display extended disk information about disks **hd3n97**, **sdfnsd**, and **hd5n98**, issue this command:

```
mmlnsd -X -d "hd3n97;sdfnsd;hd5n98"
```

The system displays information similar to:

Disk name	NSD volume ID	Device	Devtype	Node name	Remarks
hd3n97	0972846145C8E927	/dev/hdisk3	hdisk	c5n97g	server node,pr=no
hd3n97	0972846145C8E927	/dev/hdisk3	hdisk	c5n98g	server node,pr=no
hd5n98	0972846245EB501C	/dev/hdisk5	hdisk	c5n97g	server node,pr=no
hd5n98	0972846245EB501C	/dev/hdisk5	hdisk	c5n98g	server node,pr=no
sdfnsd	0972845E45F02E81	/dev/sdf	generic	c5n94g	server node
sdfnsd	0972845E45F02E81	/dev/sdm	generic	c5n96g	server node

5. The following shows the output of **mmlnsd -X** with **mmchconfig usePersistentReserve=yes**.

Disk name	NSD volume ID	Device	Devtype	Node name	Remarks
nsd0	947AC0A84F5FB55C	/dev/dm-0	dmm	c13clapv12.gpfs.net	server node,pr=yes
nsd0	947AC0A84F5FB55C	/dev/dm-19	dmm	c13clapv13.gpfs.net	server node,pr=yes
nsd0	947AC0A84F5FB55C	/dev/dm-6	dmm	c13clapv14.gpfs.net	server node,pr=yes
nsd0	947AC0A84F5FB55C	/dev/dm-15	dmm	c13clapv16.gpfs.net	server node,pr=yes
nsd1	947AC0A84F5FB564	/dev/dm-1	dmm	c13clapv12.gpfs.net	server node,pr=yes
nsd1	947AC0A84F5FB564	/dev/dm-4	dmm	c13clapv13.gpfs.net	server node,pr=yes
nsd1	947AC0A84F5FB564	/dev/dm-9	dmm	c13clapv14.gpfs.net	server node,pr=yes
nsd1	947AC0A84F5FB564	/dev/dm-13	dmm	c13clapv16.gpfs.net	server node,pr=yes

See also

- “mmcrnsd command” on page 426
- “mmdelnsd command” on page 465

Location

```
/usr/lpp/mmfs/bin
```

mmlspolicy command

Displays policy information.

Synopsis

mmlspolicy *Device* [-L]

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmlspolicy** command displays policy information for a given file system. The information displayed includes:

- When the policy file was installed.
- The user who installed the policy file.
- The node on which the policy file was installed.
- The first line of the original policy file.

For information about GPFS policies and file placement, see the Information Lifecycle Management chapter of *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system for which policy information is to be displayed. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

- L Displays the entire original policy file. If this flag is not specified, only the first line of the original policy file is displayed.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This command displays basic information for the policy installed for file system **fs2**:

```
mmlspolicy fs2
```

The system displays output similar to:

```
Policy for file system '/dev/fs2':  
  Installed by root@c103rp12.gpfs.net on Tue Mar 30 15:06:20 2010.  
  First line of policy 'policy' is:  
  /* This is the policy for the fs2 GPFS file system. */
```

2. This command displays extended information for the policy installed for file system **fs2**:

```
mmlspolicy fs2 -L
```

The system displays output similar to:

```
/* This is the policy for the fs2 GPFS file system. */

/* File Placement Rules */
RULE SET POOL 'sp4' WHERE name like '%sp4%'
RULE SET POOL 'sp5' WHERE name like '%sp5%'
RULE 'default' SET POOL 'system'

/* Exclude Rule */
RULE 'Exclude root users files' EXCLUDE WHERE USER_ID = 0 AND
name like '%org%'

/* Delete Rule */
RULE 'delete files' DELETE WHERE PATH_NAME like '%tmp%'

/* Migrate Rule */
RULE 'sp4.files' MIGRATE FROM POOL 'sp4' TO POOL 'sp5' WHERE
name like '%sp4%'

/* End of Policy */
```

3. In this example, no policy file was installed for the specified file system:

```
mmlspolicy fs4 -L
```

The system displays output similar to:

```
No policy file was installed for file system 'fs4'.
Data will be stored in pool 'system'.
```

See also

- “mmapplypolicy command” on page 266
- “mmchpolicy command” on page 391

Location

```
/usr/lpp/mmfs/bin
```

mmlspool

mmlspool command

Displays information about the known storage pools.

Synopsis

```
mmlspool Device {StoragePool[,StoragePool...] | all} [-L]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmlspool** command displays basic or detailed information about the storage pools in a file system.

Parameters

Device

Specifies the device name of the file system for which storage pool information is to be displayed. File system names do not need to be fully qualified; for example, **fs0** is as acceptable as **/dev/fs0**.

StoragePool[,StoragePool...]

Specifies one or more storage pools for which information is to be displayed.

all

Displays information about all the storage pools in specified file system.

-L Displays detailed information about each storage pool.

Exit status

0 Successful completion.

nonzero

A failure occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

If you are a nonroot user, you may specify only file systems that belong to the same cluster as the node on which the **mmlspool** command was issued.

Examples

1. To show basic information about all storage pools in a file system, issue this command:

```
mmlspool /dev/fst all
```

The system displays information similar to:

Name	Id
system	0
sataXXX	65537

2. To show more information, issue this command:

```
mmlspool /dev/fst sataXXX -L
```

The system displays information similar to:


```
sataXXX
  poolID           = 65537
  blockSize        = 256K
  usage            = dataOnly
  defaultDataReplication = 3
  maxDiskSize      = 100G
  layoutMap        = cluster
  allowWriteAffinity = yes
  writeAffinityDepth = 1
  blockGroupFactor = 2
```

See also

- “mmlsattr command” on page 519
- “mmlscallback command” on page 522
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmlsdisk command” on page 528
- “mmlspolicy command” on page 552
- “mmlsquota command” on page 559
- “mmlsnapshot command” on page 563

See also the following *IBM Spectrum Scale RAID: Administration* topics:

- “mmlsrecoverygroup command”
- “mmlsvdisk command”

Location

/usr/lpp/mmfs/bin

mmlsqos command

Displays the I/O performance values of a file system, when you enable Quality of Service for I/O operations (QoS) with the `mmchqos` command.

Synopsis

```
mmlsqos Device
      [--pool {all | Pool}]
      [--seconds Seconds]
      [--sum-classes {yes | no}]
      [--sum-nodes {yes | no}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Run the `mmlsqos` command to display the I/O performance of QoS classes. Each storage pool has two QoS classes:

- A **maintenance** class for I/O-intensive, potentially long-running GPFS commands.
- An **other** class for all other processes.

You allocate shares of IOPS to QoS classes with the `mmchqos` command.

Remember the following points:

- Allocations persist across unmounting and remounting the file system.
- QoS stops applying allocations when you unmount the file system and resumes when you remount it.
- When you change allocations or mount the file system, a brief delay due to reconfiguration occurs before QoS starts applying allocations.

For more information about this command, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Administration and Programming Reference*.

When the file system is mounted, the command displays information about the QoS classes of both explicitly named pools and *residual pools*. Residual pools are storage pools that you have not specified by name in any `mmchqos` command. When the file system is unmounted, the command displays information about only the QoS classes of explicitly named pools.

Parameters

Device

The device name of the file system to which the QoS action applies.

--pool

Display the I/O performance values for all QoS pools if **all** is specified, or for the named pool if a pool name is specified. The default is **all**.

--seconds

Display the I/O performance values for the previous number of seconds. The valid range of seconds is 1-999. The default value is 60 seconds. The values are displayed for subperiods within the period that you specify. The subperiods might be every 5 seconds over the last 60 seconds, or every 60 seconds over the last 600 seconds. You cannot configure the number or length of subperiods.

--sum-classes

Display the I/O performance for each QoS class separately if **no** is specified, or summed across all the QoS classes if **yes** is specified. The default is **no**.

--sum-nodes

If **yes** is specified, display the I/O performance summed across all the nodes in the cluster. If **no** is specified, display the I/O performance for each node separately. The default is **yes**.

Exit status

0 Successful completion.

Nonzero

A failure occurred.

Security

You must have root authority to run the **mmlsqos** command.

The node on which you enter the command must be able to execute remote shell commands on any other administration node in the cluster. It must be able to do so without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Analyzing the output from mmlsqos

```
# mmlsqos fs0 --seconds 30
QoS cfg::      enabled -- pool=system,other=inf,maintenance=400Iops
QoS status::   throttling active, monitoring active
=== for pool system
15:14:45 other iops=314.4 ioq1=4.3 qsd1=0.0 et=5
15:14:45 maint iops=215.2 ioq1=3.2 qsd1=0.0 et=5
15:14:50 other iops=308.8 ioq1=4.1 qsd1=0.0 et=5
15:14:50 maint iops=263.2 ioq1=3.7 qsd1=0.0 et=5
15:14:55 other iops=281.8 ioq1=4.0 qsd1=0.0 et=5
15:14:55 maint iops=274.8 ioq1=3.9 qsd1=0.0 et=5
15:15:00 other iops=267.0 ioq1=4.0 qsd1=0.0 et=5
15:15:00 maint iops=268.4 ioq1=3.9 qsd1=0.0 et=5
15:15:05 other iops=256.8 ioq1=4.0 qsd1=0.0 et=5
15:15:05 maint iops=252.4 ioq1=3.9 qsd1=0.0 et=5
15:15:10 other iops=245.4 ioq1=4.0 qsd1=0.0 et=5
15:15:10 maint iops=263.6 ioq1=3.9 qsd1=0.0 et=5
```

Figure 6. Sample output from **mmlsqos**

In the preceding figure, the command **mmlsqos fs0 --seconds 30** requests a display of I/O performance values over the previous 30 seconds. It requests these values for all QoS pools, for each QoS class separately, and summed across all the nodes in the cluster. The first line of output indicates that QoS is enabled, that the **other** class is set to **unlimited**, and that the **maintenance** class is set to 400 IOPS. The second line indicates that allocation is active and being monitored.

The remainder of the output shows the I/O performance values for subperiods of 5 seconds over the total period of 30 seconds. The column values are as follows:

First column

The time when the measurement period ends.

Second column

The QoS class for which the measurement is made.

iops= The performance of the class in I/O operations per second.

ioq1= The average number of I/O requests in the class that are pending for reasons other than being queued by QoS. This number includes, for example, I/O requests that are waiting for network or storage device servicing.

mmlsqos

qsd1= The average number of I/O requests in the class that are queued by QoS. When the QoS system receives an I/O request from the file system, QoS first finds the class to which the I/O request belongs. It then finds whether the class has any I/O operations available for consumption. If not, then QoS queues the request until more I/O operations become available for the class. The Qsd1 value is the average number of I/O requests that are held in this queue.

et= The interval in seconds during which the measurement was made.

You can calculate the average service time for an I/O operation as $(Ioql + Qsd1)/Iops$. For a system that is running IO-intensive applications, you can interpret the value $(Ioql + Qsd1)$ as the number of threads in the I/O-intensive applications. This interpretation assumes that each thread spends most of its time in waiting for an I/O operation to complete.

Examples

1. The following command displays the I/O performance values for all the pools in the file system over the previous 60 seconds. It does so for each QoS class separately and summed across all the nodes in the cluster.

```
mmlsqos fs0 --seconds 60
```

2. The following command displays the I/O performance values for the named pool over the previous 60 seconds. It does so for each QoS class separately and for each node separately.

```
mmlsqos fs0 --pool pname0 --sum-nodes no
```

See also

- “mmchqos command” on page 396
- “Setting the Quality of Service for I/O operations (QoS)” on page 40

Location

```
/usr/lpp/mmfs/bin
```

mmlsquota command

Displays quota information for a user, group, or fileset.

Synopsis

```
mmlsquota [-u User | -g Group] [-v | -q] [-e] [-C ClusterName]
          [--block-size {BlockSize | auto}] [Device[:Fileset] ...]
```

or

```
mmlsquota -j Fileset [-v | -q] [-e] [-C ClusterName]
          [--block-size {BlockSize | auto}] Device ...
```

or

```
mmlsquota -d {[-u] [-g] [-j]} [-C ClusterName]
          [--block-size {BlockSize | auto}] [Device ...]
```

or

```
mmlsquota -d [-C ClusterName] [--block-size {BlockSize | auto}] [Device[:Fileset] ...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

For the specified *User*, *Group*, or *Fileset* the **mmlsquota** command displays information about quota limits and current usage on each file system in the cluster. This information is displayed only if quota limits have been established and the user has consumed some amount of storage. If you want quota information for a *User*, *Group*, or *Fileset* that has no file system storage allocated at the present time, you must specify **-v**.

If neither the **-g**, **-u**, or **-j** option is specified, the default is to display only user quotas for the user who issues the command.

For each file system in the cluster, the **mmlsquota** command displays:

1. Block limits:
 - quota type (USR or GRP or FILESET)
 - current usage
 - soft limit
 - hard limit
 - space in doubt
 - grace period
2. File limits:
 - current number of files
 - soft limit
 - hard limit
 - files in doubt
 - grace period

Note: In cases where small files do not have an additional block allocated for them, quota usage may show less space usage than expected.

3. Remarks

mmlsquota

Because the sum of the *in-doubt* value and the current usage may not exceed the hard limit, the actual block space and number of files available to the user, group, or fileset may be constrained by the in-doubt value. If the in-doubt value approaches a significant percentage of the quota, run the **mmcheckquota** command to account for the lost space and files.

See “Listing quotas” on page 177 in *IBM Spectrum Scale: Administration and Programming Reference*.

This command cannot be run from a Windows node.

Parameters

-C *ClusterName*

Specifies the name of the cluster from which the quota information is obtained (from the file systems within that cluster). If **-C** is omitted, the local cluster is assumed. The cluster name specified by the **-C** flag must be part of the same multicluster group as the node issuing the **mmlsquota** command. A node that is part of a remote cluster can only see the file systems that it has been given authority to mount from the local cluster.

Device

Specifies the device name of the file system for which quota information is to be displayed. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

Fileset

Specifies the name of a fileset located on *Device* for which quota information is to be displayed.

- d** Displays the default quota limits for user, group, or fileset quotas. When specified in combination with the **-u**, **-g**, or **-j** options, default file system quotas are displayed. When specified without any of the **-u**, **-g**, or **-j** options, default fileset-level quotas are displayed.
- e** Specifies that **mmlsquota** is to collect updated quota usage data from all nodes before displaying results. If **-e** is not specified, there is the potential to display negative usage values as the quota server may process a combination of up-to-date and back-level information.
- g** *Group*
Displays quota information for the user group or group ID specified in the *Group* parameter.
- j** *Fileset*
Displays quota information for the named fileset.
- q** Prints a terse message containing information only about file systems with usage over quota.
- u** *User*
Displays quota information for the user name or user ID specified in the *User* parameter.
- v** Displays quota information on file systems where the *User*, *Group* or *Fileset* limit has been set, but the storage has not been allocated.
- block-size** {*BlockSize* | **auto**}
Specifies the unit in which the number of blocks is displayed. The value must be of the form [*n*]**K**, [*n*]**M**, [*n*]**G** or [*n*]**T**, where *n* is an optional integer in the range 1 to 1023. The default is 1K. If **auto** is specified, the number of blocks is automatically scaled to an easy-to-read value.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user:

- You may view quota information for all users, groups, and filesets.
- The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

If you are a non-root user, you may view only fileset quota information, your own quota information, and quota information for any groups to which you belong.

You must be a root user to use the **-d** option.

GPFS must be running on the node from which the **mmlsquota** command is issued.

Examples

1. Userid **paul** issued this command:

```
mmlsquota
```

The system displays information similar to:

Filesystem	type	Block Limits					File Limits				
		KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace
fsn	USR	728	100096	200192	4880	none	35	30	50	10	6days

This output shows the quotas for user **paul** in file system **fsn** set to a soft limit of 100096 KB, and a hard limit of 200192 KB. 728 KB is currently allocated to **paul**. 4880 KB is also in doubt, meaning that the quota system has not yet been updated as to whether this space has been used by the nodes, or whether it is still available. No grace period appears because the user has not exceeded his quota. If the user had exceeded the soft limit, the grace period would be set and the user would have that amount of time to bring his usage below the quota values. If the user failed to do so, the user would not be allocated any more space.

The soft limit for files (inodes) is set at 30 and the hard limit is 50. 35 files are currently allocated to this user, and the quota system does not yet know whether the 10 in doubt have been used or are still available. A grace period of six days appears because the user has exceeded his quota. The user would have this amount of time to bring his usage below the quota values. If the user fails to do so, the user is not allocated any more space.

2. To show the quotas for user **pfs001**, device **gpfs2**, and fileset **fset4**, issue this command:

```
mmlsquota -u pfs001 gpfs2:fset4
```

The system displays information similar to:

Filesystem	Fileset	type	Block Limits					File Limits					Remarks
			KB	quota	limit	doubt	grace	files	quota	limit	doubt	grace	
gpfs2	fset4	USR	4104	10240	153600	0	none	1	1000	5000	0	none	

3. To show user and group default quotas for all filesets in the **gpfs1** file system, issue this command:

```
mmlsquota -d gpfs1
```

The system displays information similar to:

Filesystem	Fileset	Default Block Limits(KB)				Default File Limits		
		type	quota	limit	quota	limit	entryType	
gpfs1	root	USR	0	0	0	0	default off	
gpfs1	root	GRP	0	0	0	0	default off	
gpfs1	fset1	USR	0	0	0	0	default off	

mmlsquota

```
gpfs1    fset1    GRP      0        0 |        0        0 default off
gpfs1    fset2    USR      0        0 |        0        0 default off
gpfs1    fset2    GRP      0        0 |        0        0 default off
```

4. To show user and group default quotas for fileset **fset1** in the **gpfs1** file system, issue this command:

```
mmlsquota -d gpfs1:fset1
```

The system displays information similar to:

Default Block Limits(KB)				Default File Limits			
Filesystem	Fileset	type	quota	limit	quota	limit	entryType
gpfs1	fset1	USR	0	0	0	0	default off
gpfs1	fset1	GRP	0	0	0	0	default off

5. To show the quotas for fileset fset0 in file system fs1, issue this command:

```
mmlsquota -j fset0 fs1 --block-size auto
```

The system displays information similar to:

Filesystem	type	Block Limits					File Limits					Remarks
		blocks	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace	
fs1	FILESET	89.25G	100G	200G	89.99M	none	13729	4000	5000	0	7 days	

See also

- “mmcheckquota command” on page 361
- “mmdefedquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmrepquota command” on page 627
- “mmquotaon command” on page 619
- “mmquotaoff command” on page 617

Location

```
/usr/lpp/mmfs/bin
```


mmlssnapshot command

Displays GPFS snapshot information.

Synopsis

```
mmlssnapshot Device [-d [--block-size {BlockSize | auto}]
  [-s {all | global | [[Fileset]:]Snapshot[[,[[Fileset]:]Snapshot...]]] | -j Fileset[[,Fileset...]]
  [--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmlssnapshot** command to display GPFS snapshot information for the specified file system or fileset. You can optionally display the amount of storage that is used by the snapshot.

Parameters

Device

The device name of the file system for which snapshot information is to be displayed. File system names do not need to be fully qualified. *fs0* is as acceptable as */dev/fs0*.

-d Displays the amount of storage that is used by the snapshot.

This operation requires an amount of time that is proportional to the size of the file system; therefore, it can take several minutes or even hours on a large and heavily-loaded file system.

This optional parameter can impact overall system performance. Avoid running the **mmlssnapshot** command with this parameter frequently or during periods of high file system activity.

--block-size {*BlockSize* | **auto**}

Specifies the unit in which the number of blocks is displayed. The value must be of the form [*n*]**K**, [*n*]**M**, [*n*]**G** or [*n*]**T**, where *n* is an optional integer in the range 1 - 1023. The default is 1 K. If **auto** is specified, the number of blocks is automatically scaled to an easy-to-read value.

-s Displays the attributes for the specified snapshots.

all

Displays information for all snapshots. This option is the default.

global

Displays information for global snapshots.

[[*Fileset*]:]

:

A colon (:) followed by a snapshot name indicates a global snapshot. For example, **:SS01** indicates a global snapshot with the name **SS01**. If a global snapshot with that name exists, the command displays information about it.

Fileset:

A fileset name followed by a colon (:) followed by a snapshot name indicates a fileset snapshot. For example, **fset02:SS01** indicates a snapshot of fileset **fset02** with the name **SS01**. If a snapshot of the fileset with that snapshot name exists, then the command displays information about it.

***Snapshot*[[,*Snapshot*...]]**

Displays information for the specified snapshots.

-j *Fileset*[[,*Fileset*...]]

Displays only snapshots that contain the specified filesets; including all global snapshots.

mmlssnapshot

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure occurred.

Security

You must be a root user or fileset owner to use the **-d** parameter.

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

If you are a non-root user, you can specify only file systems that belong to the same cluster as the node on which the **mmlssnapshot** command was issued.

Examples

The following command displays information about all the existing snapshots in file system fs1:

```
mmlssnapshot fs1
```

The following command displays information about snapshots named SS01. The snapshots can be global snapshots or fileset snapshots:

```
mmlssnapshot fs1 -s SS01
```

The following command displays information about a global snapshot named gSS01:

```
mmlssnapshot fs1 -s :gSS01
```

A snapshot name can contain a colon (:). The following command displays information about a global snapshot named :gSS04:

```
mmlssnapshot fs1 -s ::gSS04
```

The following command displays information about a fileset snapshot with the name fsSS01 that is a snapshot of fileset fset02:

```
mmlssnapshot fs1 -s fset02:fsSS01
```

The following command displays information about global snapshots with the names gSS01 and gSS02 and a fileset snapshot of fileset fset02 named fsSS02:

```
mmlssnapshot fs1 -s :gSS01,:gSS02,fset02:fsSS02
```

The following command displays information about global snapshots and fileset snapshots that contain the filesets fset02 and fset03:

```
mmlssnapshot fs1 -j fset02,fset03
```

See also

- “mmcrsnapshot command” on page 431
- “mmdelsnapshot command” on page 467
- “mmrestorefs command” on page 635
- “mmsnapdir command” on page 674

Location

```
/usr/lpp/mmfs/bin
```

mmmigratefs

mmmigratefs command

Performs needed conversions to support new file system features.

Synopsis

```
mmmigratefs Device [--fastea] [--online | --offline]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmmigratefs** command to enable features that require existing on-disk data structures to be converted to a new format.

Before issuing the **mmmigratefs** command, see the topic about migration, coexistence, and compatibility in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*. You must ensure that all nodes in the cluster have been migrated to the latest level of GPFS code and that you have successfully run the **mmchconfig release=LATEST** command. You must also ensure that the new features have been enabled by running **mmchfs -V full**.

The **mmmigratefs** command can be run with the file system mounted or unmounted. If **mmmigratefs** is run without the **--online** or **--offline** parameters specified, the command will determine the mount status of the file system and run in the appropriate mode.

Parameters

Device

The device name of the file system to be migrated. File system names need not be fully qualified; for example, **fs0** is just as acceptable as **/dev/fs0**. This must be the first parameter.

--fastea

Convert the existing extended attributes to the new format required for storing the attributes in the file's inode and thereby allowing for faster extended-attribute access.

--online

Allows the **mmmigratefs** command to run while the file system is mounted.

--offline

Allows the **mmmigratefs** command to run while the file system is unmounted.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmmigratefs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To enable fast extended attribute access for file system fs3, issue this command:

```
mmmigratefs fs3 --fastea
```

The system displays information similar to the following:

```
Enabling fastea support
 11.19 % complete on Thu Nov 14 13:50:24 2013 ( 167936 inodes      328 MB)
 33.21 % complete on Thu Nov 14 13:51:56 2013 ( 498260 inodes      973 MB)
100.00 % complete on Thu Nov 14 13:52:04 2013
Finalizing upgrade
 11.19 % complete on Thu Nov 14 13:52:27 2013 ( 167936 inodes      328 MB)
 26.78 % complete on Thu Nov 14 13:53:07 2013 ( 401834 inodes      785 MB)
100.00 % complete on Thu Nov 14 13:53:27 2013
Feature 'fastea' is now enabled on "fs3".
```

See also

- “mmchconfig command” on page 331
- “mmchfs command” on page 371

Location

```
/usr/lpp/mmfs/bin
```

mmmount command

Mounts GPFS file systems on one or more nodes in the cluster.

Synopsis

```
mmmount {Device | DefaultMountPoint | DefaultDriveLetter |  
  all | all_local | all_remote | {-F DevceFileName}}  
  [-o MountOptions] [-a | -N {Node[,Node...]} | NodeFile | NodeClass]
```

or

```
mmmount Device {MountPoint | DriveLetter}  
  [-o MountOptions] [-a | -N {Node[,Node...]} | NodeFile | NodeClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmmount** command mounts the specified GPFS file system on one or more nodes in the cluster. If no nodes are specified, the file systems are mounted only on the node from which the command was issued. A file system can be specified using its device name or its default mount point, as established by the **mmcrfs**, **mmchfs** or **mmremotefs** commands.

When **all** is specified in place of a file system name, all GPFS file systems will be mounted. This also includes remote GPFS file systems to which this cluster has access.

Parameters

Device | *DefaultMountPoint* | *DefaultDriveLetter* | **all** | **all_local** | **all_remote** | {-F *DeviceFileName*}

Indicates the file system or file systems to be mounted.

Device

The device name of the file system to be mounted. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

DefaultMountPoint

The mount point associated with the file system as a result of the **mmcrfs**, **mmchfs**, or **mmremotefs** commands.

DefaultDriveLetter

The Windows drive letter associated with the file system as a result of the **mmcrfs** or **mmchfs** command.

all

Indicates all file systems known to this cluster.

all_local

Indicates all file systems owned by this cluster.

all_remote

Indicates all files systems owned by another cluster to which this cluster has access.

-F *DeviceFileName*

Specifies a file containing the device names, one per line, of the file systems to be mounted.

This must be the first parameter.

DriveLetter

The location where the file system is to be mounted. If not specified, the file system is mounted at its

default drive letter. This option can be used to mount a file system at a drive letter other than its default one or to mount a file system that does not have an established default drive letter.

MountPoint

The location where the file system is to be mounted. If not specified, the file system is mounted at its default mount point. This option can be used to mount a file system at a mount point other than its default mount point.

Options

-a Mount the file system on all nodes in the GPFS cluster.

-N {*Node* [,*Node*...] | *NodeFile* | *NodeClass*}

Specifies the nodes on which the file system is to be mounted.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

-o *MountOptions*

Specifies the mount options to pass to the mount command when mounting the file system. For a detailed description of the available mount options, see “GPFS-specific mount options” on page 28 in *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmmount** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To mount all GPFS file systems on all of the nodes in the cluster, issue this command:

```
mmmount all -a
```

2. To mount file system **fs2** read-only on the local node, issue this command:

```
mmmount fs2 -o ro
```

3. To mount file system **fs1** on all NSD server nodes, issue this command:

```
mmmount fs1 -N nsdsnodes
```

See also

- “mmumount command” on page 684
- “mmlsmount command” on page 544

Location

/usr/lpp/mmfs/bin

mmnfs command

Manages NFS exports and configuration.

Synopsis

```
mmnfs export add Path [--client ClientOptions]
```

or

```
mmnfs export remove Path [--force]
```

or

```
mmnfs export change Path [--nfsadd ClientOptions]  
[--nfschange ClientOptions]
```

or

```
mmnfs export list [--nfsdefs Path] [--raw]
```

or

```
mmnfs export load ExportCFGFile
```

or

```
mmnfs configuration list [--exportdefs] [--raw]
```

or

```
mmnfs configuration change "Option=Value:Option=Value1,Value2:Option=Value..."
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmnfs export** commands to add, change, list, load, or remove NFS export declarations for IP addresses on nodes that are configured as CES types.

Use the **mmnfs configuration** commands to list and change NFS configuration.

The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

Parameters

export

Manages the NFS export configuration for the cluster with one of the following actions:

add

Creates a new configuration file for the NFS server in case it does not yet exist. If there is already an export configuration file, then it is extended with the provided additional export parameters. This export configuration file is used by the NFS server to create an NFS export for the given *Path* so that clients can connect to it. If there is already an existing export for the given *Path* then an error is shown. Each export configuration set has internally its own unique identifier number. This number is automatically incremented for each added export. The **mmnfs export add** command attempts to add the new export also to running NFS server instances, and may fail if

one or more instances are not running. This is not a critical issue, since the configuration changes have been made in the repository and will be applied later when restarting the NFS server instance.

The authentication method must be established before an NFS export can be defined.

The export *Path* must be an existing path in the GPFS file system, which is intended to be exported or is already exported to external clients using the NFS protocol.

Note: The paths which are not within the GPFS file system cannot be exported using the commands.

Creating nested exports (such as */path/to/folder* and */path/to/folder/subfolder*) is strongly discouraged since this may lead to serious issues in data consistency. Be very cautious when creating and using nested exports.

Some export configuration commands may allow multiple client declarations, and therefore they have separators to distinguish them.

The following separators can be used:

- **Colon** to separate multiple allowed values for a given attribute. For example, the key/value pair "Protocols=3:4" allows the NFS protocols v3 and v4 to be declared for an export.
- **Comma** to separate key/value pairs within a client declaration list and a **Semicolon** to separate client declaration lists.

For example:

```
--client 192.0.2.0/20 (Access_Type=RW, Protocols=3:4);198.51.100.0/20
(Access_Type=R0,Protocols=3:4,Transports=TCP:UDP)
```

Note: To take advantage of the GPFS independent fileset features such as quotas, snapshots, and data management, the export paths can be made from GPFS filesets (either dependent or independent).

--client *ClientOptions*

Declares the client specific settings. *ClientOptions* can be a list of one or more client definitions. It is advised to quote the argument list to avoid a wrong parsing by the interpreter. For a list of client definitions that can be specified with the **--client** option, see *List of supported client options for the mmnfs export {add | change} command*.

remove

Removes the requested export from the configuration file and also from running NFS server instances, and may fail if one or more instances are not running.

Note: This command does not remove data.

The **--force** option is currently not used.

change

Modifies an existing export configuration for the export specified by *Path*, if the export exists. If the export does not exist, then an error is shown.

Note: Only client-related attributes are modified by this command, but not the basic export settings such as path.

The **--nfsposition** flag can be only used together with either **--nfsadd** or **--nfschange**. It cannot be used standalone or together with **--nfsremove**. When **--nfsadd** and **--nfsremove** are given on the same command line, then the remove procedure is executed first internally.

Note: An export change will cause a restart of the NFS server.

mmnfs

--nfsadd *ClientOptions*

Adds a new client declaration for the specified *Path*. *ClientOptions* can be a list of one or more client definitions. It is advised to quote the argument list to avoid a wrong parsing by the interpreter. For a list of client definitions that can be specified with the **--client** option, see *List of supported client options for the mmnfs export {add | change} command*.

list

Lists the declared exports based on the entries in the configuration file stored in the repository. The sequence of rows in the output for a given path reflects also the sequence of the internal client declaration list for each exported path. The sequence of client declarations within an export can be reordered using the **mmnfs export change** *Path* with the **--nfschange** and **--nfsposition** options. The output can be formatted human readable (default) or machine readable.

--nfsdefs *Path*

Lists the export configuration details for the specified *Path*. Without this option the **mmnfs export list** command shows a table with some basic configuration settings for all declared exports.

--raw

The output of the **mmnfs export list** command can be in a tabular form (human readable, default) or in a list of colon separated values in a machine readable format. Use the **--raw** option to create the machine readable output.

load

Overwrites (deletes) all existing NFS export declarations in the repository, if any. The export declarations are fetched from a file provided to the load operation, which could contain a larger number of export declarations. Some basic format checks are done during export load. After loading export declarations from a file, the NFS service is restarted across all the nodes in the cluster.

ExportCFGFile

The file name for the new exports declarations. This file is loaded and stored in the repository to be published on all CES nodes running the NFS server. This load procedure can be used to load a set of export declarations and that will remove any previous configuration. The NFS servers are restarted in order to apply the changes.

List of supported client options for the mmnfs export {add | change} command:

ACCESS_TYPE

Allowed values are none, RW, RO, MDONLY, and MDONLY_RO. The default value is none.

PROTOCOLS

Allowed values are 3, 4, NFS3, NFS4, V3, V4, NFSv3, and NFSv4. The default value is 3,4.

TRANSPORTS

Allowed values are TCP and UDP. The default value is TCP.

ANONYMOUS_UID

Allowed values are between -2147483648 and 4294967295. The default value is -2.

ANONYMOUS_GID

Allowed values are between -2147483648 and 4294967295. The default value is -2.

SECTYPE

Allowed values are none, sys, krb5, krb5i, and krb5p. The default value is sys.

PRIVILEGEDPORT

Allowed values are true and false. The default value is false.

MANAGE_GIDS

Allowed values are true and false. The default value is false.

SQUASH

Allowed values are root, root_squash, all, all_squash, allsquash, no_root_squash, none, and noidsquash. The default value is root_squash.

NFS_COMMIT

Allowed values are true and false. The default value is false.

Important: Use NFS_COMMIT very carefully because it changes the behavior of how transmitted data is committed on the server side to a NFS v2 like sync-mode on every write action.

CLIENTS

Allowed values are IP addresses in IPv4 or IPv6 notations, or * for all. The default value is *.

configuration

Manages NFS configuration for a CES cluster:

list

Displays the NFS configuration parameters and their values. This command also displays all the default export configurations. This is used as the defaults by the **mmnfs export add** command if no other client attributes are specified. The output can be formatted to be human readable or machine readable.

--exportdefs

If this option is specified, the command displays the default export configuration parameters.

--raw

The command output can be in a tabular form (human readable, default) or in a list of colon separated values in a machine readable format. Use the **--raw** option to create the machine readable output.

change

Modifies the NFS configuration parameters. NFS is restarted across all the nodes on which NFS is running, when this command is executed. Only some configuration options can be modified by this command.

The configuration options that can be modified and their allowed values are as follows:

NFS_PROTOCOLS

Allowed values are 3, 4, NFS3, NFS4, V3, V4, NFSv3, and NFSv4. The default value is 3,4.

NFS_PORT

Specifies the port where the NFS server will start. Allowed values are between 0 and 65535. The default value is 0.

MNT_PORT

Specifies the port for the NFSv3 Mount protocol. Allowed values are between 0 and 65535. The default value is 0.

NLM_PORT

Specifies the NLM port for NFSv3. Allowed values are between 0 and 65535. The default value is 0.

RQUOTA_PORT

Specifies the RQUOTA port for NFSv3. Allowed values are between 0 and 65535. The default value is 0.

STATD_PORT

Specifies the STATD port for NFSv3. Allowed values are between 0 and 65535. The default value is 0.

SHORT_FILE_HANDLE

Allowed values are True or False. The default value is False.

Set this flag to True when using VMware NFS clients.

mmnfs

LEASE_LIFETIME

Allowed values are between 0 and 120. The default value is 60.

DOMAIN_NAME

String. The default value is "localdomain".

IDMAPD_DOMAIN

Domain name in ID Mapd,String. The default value is localdomain.

LOCAL_REALMS

Local Realm in ID Mapd configuration. The default value is localdomain.

LOG_LEVEL

Allowed values are NULL, FATAL, MAJ, CRIT, WARN, EVENT, INFO , DEBUG, MID_DEBUG, and FULL_DEBUG. The default value is EVENT.

ENTRIES_HWMARK

The high water mark for NFS cache entries. Beyond this point, NFS will try to evict some objects from its cache. The default is 1500000.

Note: Specifying a port number in the NFS service configuration with the value of '0' means that the service is picking a port number dynamically. This port number might change across service restarts. If a firewall is to be established between the NFS server and the NFS clients, specific port number can be configured via the command to establish discrete firewall rules. Note that the NFS_PORT 2049 is a well known and established convention as NFS servers and clients typically expect this port number. If the NFS_PORT is changed to another value, ensure that the clients are also informed accordingly about this port value (depending on the client platform and type of client). Changing the NFS service port numbers impacts existing clients and a remount of the client is required.

The export defaults that can be set are:

ACCESS_TYPE

Allowed values are none, RW, RO, MDONLY, and MDONLY_RO. The default value is none.

Note: Changing this option to any value other than none will expose data to all NFS clients that can access your network, even if the export is created using **add --client ClientOptions** to limit that clients access. All clients, even if not declared with the **--client** in the **mmnfs export add**, will have access to the data. The global value will apply to an unseen *, even if **showmount -e CESIP** does not display it. Use caution if you change it in this global definition.

ANONYMOUS_UID

Allowed values are between -2147483648 and 4294967295. The default value is -2.

ANONYMOUS_GID

Allowed values are between -2147483648 and 4294967295. The default value is -2.

MANAGE_GIDS

Allowed values are true and false. The default value is false.

NFS_COMMIT

Allowed values are true and false. The default value is false.

Important: Use NFS_COMMIT very carefully because it changes the behavior of how transmitted data is committed on the server side to a NFS v2 like sync-mode on every write action.

PRIVILEGEDPORT

Allowed values are true and false. The default value is false.

PROTOCOLS

Allowed values are 3, 4, NFS3, NFS4, V3, V4, NFSv3 , and NFSv4. The default value is 3,4.

SECTYPE

Allowed values are none, sys, krb5, krb5i, and krb5p. The default value is sys.

SQUASH

Allowed values are root, root_squash, all, all_squash, allsquash, no_root_squash, none, and noidsquash. The default value is root_squash.

Note: Changing this option to no_root_squash will expose data to root on all NFS clients, even if the export is created using `add --client ClientOptions` to limit the root access of that client.

TRANSPORTS

Allowed values are TCP and UDP. The default value is TCP.

If export defaults are set, then new exports that are created will pick up the export default values.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the `mmnfs` command.

The node on which the command is issued must be able to execute remote shell commands on any other CES node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system in IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To create an NFS export (using a netgroup), issue this command:

```
mmnfs export add /mnt/gpfs0/netgrouppath \
--client "@netgroup(Access_Type=R0,Squash=allsquash)"
```

The system displays output similar to this:

The NFS export was created successfully.

Note: Instead of a netgroup, a client IP address can also be declared, like `--client "1.2.3.4"`.

2. To create an NFS export (using a client IP), issue this command:

```
mmnfs export add /mnt/gpfs0/netgrouppath --client "192.0.2.0/20 (Access_Type=RW)"
```

The system displays output similar to this:

The NFS export was created successfully.

3. To add a client definition, issue these commands:

```
mmnfs export list --nfsdefs /gpfs/fs1/export_1
```

The system displays output similar to this:

Path	Delegations	Clients	Access_Type	Protocols	Transports	Squash	Anonymous_uid	Anonymous_gid	SecType	PrivilegedPort	Export_id	DefaultDelegation	Manage_Gids	NFS_Commit
/gpfs/fs1/export_1	none	192.0.2.8	R0	3,4	TCP	ROOT_SQUASH	-2	-2	sys	false	3	none	false	false

Now add a client definition that will be more restrictive to a different client, 198.51.100.10, by issuing the command:

```
mmnfs export change /gpfs/fs1/export_1 \
--nfsadd "198.51.100.10 (Access_Type=MDONLY,Squash=allsquash)"
```

mmnfs

```
mmnfs export list
```

The system displays output similar to this:

Path	Delegations	Clients
/gpfs/fs1/controller	none	*
/gpfs/fs1/export_1	none	192.0.2.8
/gpfs/fs1/export_1	none	198.51.100.10

Now add a client definition that will be very permissive but we want it to be last on the list so that the more restrictive attributes for client 192.0.2.8 will take precedence for that one client, by issuing the command:

```
mmnfs export change /gpfs/fs1/export_1 \  
--nfsadd "192.0.2.0/20(Access_Type=RW,Squash=no_root_squash)" --nfsposition 4  
mmnfs export list --nfsdefs /gpfs/fs1/export_1
```

The system displays output similar to this:

Path	Delegations	Clients	Access _Type	Protocols	Transports	Squash	Anonymous _uid	Anonymous _gid	SecType	Privileged Port	Export_id	Default Delegation	Manage_Gids	NFS_Commit
/gpfs/fs1/export_1	none	192.0.2.8	RO	3,4	TCP	ROOT_SQUASH	-2	-2	sys	false	3	none	false	false
/gpfs/fs1/export_1	none	198.51.100.10	MDONLY	3,4	TCP	allsquash	-2	-2	sys	false	3	none	false	false
/gpfs/fs1/export_1	none	192.0.2.0/20	RW	3,4	TCP	no_root_squash	-2	-2	sys	false	3	none	false	false

Note: `mmnfs export change` will restart NFS services on all CES nodes in the cluster.

4. To remove an NFS export, issue this command:

```
mmnfs export change /mnt/gpfs0/somepath --nfsremove "1.2.3.1"
```

The system displays output similar to this:

```
[ OK ].12: Stopping ganesha.nfsd: [ OK ]  
[ OK ].11: Stopping ganesha.nfsd: [ OK ]  
203.0.113.12: Starting ganesha.nfsd: [ OK ]  
203.0.113.11: Starting ganesha.nfsd: [ OK ]
```

Note: This command only removes a single client definition of the export. It does not remove the NFS export. It removes the client definition for the IP "1.2.3.1."

5. To modify an NFS export, issue this command:

```
mmnfs export change /mnt/gpfs0/p1 \  
--nfschange "203.0.113.2(access_type=R0)" --nfsposition "*"
```

The system displays output similar to this:

```
[ OK ].12: Stopping ganesha.nfsd: [ OK ]  
[ OK ].11: Stopping ganesha.nfsd: [ OK ]  
203.0.113.12: Starting ganesha.nfsd: [ OK ]  
203.0.113.11: Starting ganesha.nfsd: [ OK ]
```

6. To list NFS exports, issue this command:

```
mmnfs export list
```

The system displays output similar to this:

Path	Delegations	Clients
/gpfs/fs1/controller	none	*
/gpfs/fs1/export_1	none	*
/gpfs/fs1/export_2	none	*

7. To list NFS exports, issue this command:

```
mmnfs export list --nfsdefs /mnt/gpfs0/p1
```

The system displays output similar to this:

Path	Delegations	Clients	Access_ Type	Protocols	Transports	Squash	Anonymous_ uid	Anonymous_ gid	SecType	Privileged Port	Export_ id	Default Delegation
/mnt/gpfs0/p1	none	203.0.113.2	RO	3,4	TCP	root_squash	-2	-2	sys	false	1	none
/mnt/gpfs0/p1	none	*	RW	3,4	TCP	root_squash	-2	-2	sys	false	1	none
/mnt/gpfs0/p1	none	203.0.113.1	RO	3,4	TCP	root_squash	-2	-2	sys	false	1	none

8. To list all NFS configuration, issue this command:

```
mmnfs configuration list
```

The system displays output similar to this:

```
NFS Ganesha Configuration:
```

```
=====
```

```
NFS_PROTOCOLS: 3,4
```

```
NFS_PORT: 2049
```

```
MNT_PORT: 0
```

```
NLM_PORT: 0
```

```
RQUOTA_PORT: 0
```

```
LEASE_LIFETIME: 60
```

```
DOMAINNAME: VIRTUAL1.COM
```

```
DELEGATIONS: Disabled
```

```
=====
```

```
STATD Configuration
```

```
=====
```

```
STATD_PORT: 0
```

```
=====
```

```
Export Defaults
```

```
=====
```

```
ACCESS_TYPE: NONE
```

```
PROTOCOLS: 3,4
```

```
TRANSPORTS: TCP
```

```
ANONYMOUS_UID: -2
```

```
ANONYMOUS_GID: -2
```

```
SECTYPE: SYS
```

```
PRIVILEGEDPORT: FALSE
```

```
MANAGE_GIDS: FALSE
```

```
SQUASH: ROOT_SQUASH
```

```
NFS_COMMIT: FALSE
```

```
Log Configuration
```

```
=====
```

```
DEFAULT_LOG_LEVEL: EVENT
```

9. To change STATD_PORT configuration, issue this command (When a port is assigned, STATD is started on the given port):

```
mmnfs configuration change STATD_PORT=32765
```

The system displays output similar to this:

```
NFS Ganesha Configuration:
```

```
=====
```

```
NFS_PROTOCOLS: 3,4
```

```
NFS_PORT: 2049
```

```
MNT_PORT: 0
```

```
NLM_PORT: 0
```

```
RQUOTA_PORT: 0
```

```
LEASE_LIFETIME: 60
```

```
DOMAINNAME: VIRTUAL1.COM
```

```
DELEGATIONS: Disabled
```

```
=====
```

```
STATD Configuration
```

```
=====
```

```
STATD_PORT: 32765
```

```
=====
```

mmnfs

```
Export Defaults
=====
ACCESS_TYPE: NONE
PROTOCOLS: 3,4
TRANSPORTS: TCP
ANONYMOUS_UID: -2
ANONYMOUS_GID: -2
SECTYPE: SYS
PRIVILEGEDPORT: FALSE
MANAGE_GIDS: FALSE
SQUASH: ROOT_SQUASH
NFS_COMMIT: FALSE
```

```
Log Configuration
=====
DEFAULT_LOG_LEVEL: EVENT
```

Note: The **mmnfs** command has an interactive mode that provides some prompting as follows:

```
mmnfs
mmnfs [ -I ] Command
mmnfs -I
mmnfs -I>
```

See also

- “mmces command” on page 304
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmobj command” on page 581
- “mmsmb command” on page 663
- “mmuserauth command” on page 690

Location

```
/usr/lpp/mnfs/bin
```


mmnsddiscover command

Rediscovered paths to the specified network shared disks.

Synopsis

```
mmnsddiscover [-a | -d "Disk[;Disk...]" | -F DiskFile] [-C ClusterName]
              [-N {Node[,Node...] | NodeFile | NodeClass}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmnsddiscover** command is used to rediscover paths for GPFS NSDs on one or more nodes. If you do not specify a node, GPFS rediscovers NSD paths on the node from which you issued the command.

On server nodes, **mmnsddiscover** causes GPFS to rediscover access to disks, thus restoring paths which may have been broken at an earlier time. On client nodes, **mmnsddiscover** causes GPFS to refresh its choice of which NSD server to use when an I/O operation occurs.

In general, after the path to a disk is fixed, the **mmnsddiscover** command must be first run on the server that lost the path to the NSD. After that, run the command on all client nodes that need to access the NSD on that server. You can achieve the same effect with a single **mmnsddiscover** invocation if you utilize the **-N** option to specify a node list that contains all the NSD servers and clients that need to rediscover paths.

Parameters

- a** Rediscovered paths for all NSDs. This is the default.
- d "DiskName[;DiskName]"**
Specifies a list of NSDs whose paths are to be rediscovered.
- F DiskFile**
Specifies a file that contains the names of the NSDs whose paths are to be rediscovered.
- C ClusterName**
Specifies the name of the cluster to which the NSDs belong. This defaults to the local cluster if not specified.
- N {Node[,Node...] | NodeFile | NodeClass}**
Specifies the nodes on which the rediscovery is to be done.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

- 0** Successful completion.
- nonzero**
A failure has occurred.

Security

You must have root authority to run the **mmnsddiscover** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For

mmnsddiscover

more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To rediscover the paths for all of the NSDs in the local cluster on the local node, issue the command:
`mmnsddiscover`

The system displays output similar to:

```
mmnsddiscover: Attempting to rediscover the disks.  
This may take a while ...
```

```
mmnsddiscover: Finished.
```

2. To rediscover the paths for all of the NSDs in the local cluster on all nodes in the local cluster, issue the command:

```
mmnsddiscover -a -N all
```

The system displays output similar to:

```
mmnsddiscover: Attempting to rediscover the disks.  
This may take a while ...
```

```
mmnsddiscover: Finished.
```

3. To rediscover the paths for a given list of the NSDs on a node in the local cluster, issue the command:
`mmnsddiscover -d "gpfs1nsd;gpfs2nsd" -N c6f2c2vp5`

The system displays output similar to:

```
mmnsddiscover: Attempting to rediscover the disks. This may take a while ...  
c6f2c2vp5.ppd.pok.ibm.com: GPFS: 6027-1805 [N] Rediscovered nsd server access to gpfs1nsd.  
c6f2c2vp5.ppd.pok.ibm.com: GPFS: 6027-1805 [N] Rediscovered nsd server access to gpfs2nsd.  
mmnsddiscover: Finished.
```

See also

- “mmchnsd command” on page 388
- “mmcrnsd command” on page 426
- “mmdeinsd command” on page 465
- “mmlnsd command” on page 549

Location

/usr/lpp/mmfs/bin

mmobj command

Manages configuration of Object protocol service, and administers storage policies for object storage, unified file and object access, and multi-region object deployment.

Synopsis

```
mmobj swift base -g GPFSMountPoint --cluster-hostname CESHostName
    [-o ObjFileset] [-i MaxNumInodes]
    {{{--local-keystone [--db-password Password] [--admin-token Token]}
    | [--remote-keystone-url URL] [--configure-remote-keystone]}
    --admin-password Password [--admin-user AdminUser]
    [--swift-user SwiftUser] [--swift-password SwiftPassword]
    [--enable-file-access] [--enable-s3] [--enable-multi-region]
    [--join-region-file RegionFile] [--region-number RegionNumber]

or

mmobj config list --ccrfile CCRFile [--section Section [--property PropertyName]]

or

mmobj config change --ccrfile CCRFile --section Section --property PropertyName --value Value

or

mmobj config change --ccrfile CCRFile --merge-file MergeFile

or

mmobj policy list [--policy-name PolicyName] | [--policy-function PolicyFunction]] [--verbose]

or

mmobj policy create PolicyName [-f FilesetName] [-i MaxNumInodes]
    [--enable-compression --compression-schedule CompressionSchedule]
    [--enable-file-access]

or

mmobj policy change PolicyName --default

or

mmobj policy change PolicyName --deprecate State

or

mmobj policy change --add-local-region

or

mmobj policy change --remove-region-number RegionNumber

or

mmobj file-access {--object-path ObjectPath | --storage-policy PolicyName
    [--account-name AccountName [--container-name ContainerName]
    [--object-name ObjectName]]} [--node NodeName]

or

mmobj multiregion list

or

mmobj multiregion enable
```

mmobj

or

```
mmobj multiregion export --region-file RegionFile
```

or

```
mmobj multiregion import --region-file RegionFile
```

or

```
mmobj multiregion remove --region-number RegionNumber --force
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmobj** command to modify and change the Object protocol service configuration, and to administer storage policies for object storage, unified file and object access, and multi-region object deployment.

Note: The **mmobj config list** and the **mmobj config change** commands are used to list and change the configuration values for the underlying Swift service stored in the Cluster Configuration Repository (CCR).

At least one CES IP address is required in the node running the **mmobj swift base** command to set `object_singleton_node` and `object_database_node` attributes.

- The node with the `object_database_node` attribute runs the keystone database.
- The node with the `object_singleton_node` attribute runs unique object services across the CES cluster.

You can verify the address using the **mmces address list**. The IP address can be added using the **mmces address add** command.

In case of a complete reconfiguration, use the **mmces service disable OBJ** command to perform necessary cleanup before running the **mmobj swift base** command again.

Object authentication can be either local or remote. If the Object authentication is local, the Keystone identity service and the Keystone database will be running in and handled by the CES cluster. If the Object authentication is remote, the Keystone server must be fully configured and running before Object services are installed.

Parameters

swift

Configures the underlying Swift services:

base

Specifies the configuration of the Object protocol service.

-g *GPFSMountPoint*

Specifies the mount path of the GPFS file system used by Swift.

GPFSMountPoint is the mount path of the GPFS file system used by the Object store.

--cluster-hostname *CESHostName*

Specifies the hostname which is used to return one of the CES IP addresses. The returned CES IP address is used in the endpoint for the identity and Object-store values stored in Keystone.

CESHostName is the value for cluster host name used in the identity and object-store endpoint definitions in Keystone. Ideally, it should be a hostname which will return one the CES IP

addresses, such as a round-robin DNS. It could also be a fixed IP of a load balancer that distributes requests to one of the CES nodes. It is not recommended to use an ordinary CES IP since all identity and object-store requests would be routed to the single node with that address and may cause performance issues.

--local-keystone

Specifies that a new Keystone server will be installed and configured locally in the cluster.

--db-password *Password*

Specifies the password for the 'keystone' user in the postgres database. Defaults to the value for `--admin-password`.

--admin-user *User*

Specifies the name of the admin user in Swift. The default value is `admin`.

--admin-password *Password*

Specifies the password to be used when creating the administrator user in Keystone.

--admin-token *Token*

Specifies the admin token to be used for the initial Keystone setup. If it is not specified, a random string will be used.

--swift-user *User*

Specifies the user for the Swift services. The default value is `swift`.

--swift-password *Password*

Specifies the password for the Swift user. The default value is the password value from `--admin-password`.

--remote-keystone-url *URL*

Specifies the URL to an existing Keystone service.

--configure-remote-keystone

When a remote Keystone is used, this specifies that the remote Keystone should be configured as necessary. The required users, roles and endpoints needed by the Swift services will be added to the Keystone server. Keystone authentication information needs to be specified with the `--admin-password` or the `--admin-token` flag to enable the configuration. If this flag is not specified, the remote Keystone is not modified and the administrator will need to add the appropriate entries for the Swift configuration after the install is complete.

--admin-password *Password*

Specifies the password to be used when creating the administrator user in Keystone.

-o *ObjFileset*

Specifies the name of the fileset to be created in GPFS for the object storage.

ObjFileset is the name of the independent fileset that will be created for the Object store. By default, `object_fileset` is created.

-i

Specifies the maximum number of inodes for the Object fileset.

MaxNumInodes

The maximum number of inodes for the Object fileset. By default, 8000000 is set.

--enable-s3

Sets the s3 capability (Amazon S3 emulation support) to true. By default, S3 emulation is not enabled.

--enable-file-access

Sets the file-access capability initially to true. Further configuration is still necessary using the `mmobj file-access` command. By default, the file-access capability is not enabled.

mmobj

--enable-multi-region

Sets the multi-region capability initially to true. By default, multi-region capability is not enabled.

--join-region-file *RegionFile*

Specifies that this object installation will join an existing object multi-region Swift cluster. *RegionFile* is the region data file created by the **mmobj multiregion export** command from the existing multi-region cluster.

Note: The use of the **--configure-remote-keystone** flag is recommended so that the region-specific endpoints for this region are automatically created in Keystone.

--region-number *RegionNumber*

Specifies the Swift region number for this cluster. If it is not specified, the default value is to 1. In a multi-region configuration, this flag is required and must be a unique region number which is not used by another region in the multi-region environment.

config

Administers the Object configuration:

list

Lists configuration values of the underlying Swift/Keystone service stored in the CCR.

--section *Section*

Retrieves values for the specified section only.

The section is the heading enclosed in brackets ([]) in the associated configuration file.

--property *PropertyName*

Retrieves values for the specified property only.

change

Enables modifying Swift/Keystone configuration files. After you modify the configuration files, the CES monitoring framework downloads them from the CCR and distributes them to all the CES nodes in the cluster. The framework also automatically restarts the services which depend on the modified configuration files.

Note: It is recommended to not directly modify the configuration files in `/etc/swift` and `/etc/keystone` folders as they can be overwritten at any time by the files stored in the CCR.

--section *Section*

Specifies the section in the file that contains the parameter.

The section is the heading enclosed in brackets ([]) in the associated configuration file.

--property *PropertyName*

Specifies the name of the property to be set.

--value *NewValue*

Specifies the value of the *PropertyName*.

--merge-file *MergeFile*

Specifies a file in the openstack-config .conf format that contains multiple values to be changed in a single operation. The properties in *MergeFile* can represent new properties or properties to be modified. If a section or property name in *MergeFile* begins with a '-' character, that section or property is deleted from the file. For example, a *MergeFile* with the following contents would delete the ldap section, set the connections property to 512, and delete the **noauth** property from the database section.

```
[-ldap]
[database]
connections = 512
-noauth =
```

Parameter common for both mmobj config list and mmobj config change commands:

--ccrfile *CCRFile*

Indicates the name of the Swift, Keystone, or object configuration file stored in the CCR.

Some of the configuration files stored in the Cluster Configuration Repository (CCR) are:

- account-server.conf
- container-reconciler.conf
- container-server.conf
- object-expirer.conf
- object-server.conf
- proxy-server.conf
- swift.conf
- keystone.conf
- keystone-paste.ini
- spectrum-scale-object.conf
- object-server-sof.conf
- spectrum-scale-objectizer.conf

policy

Administers the storage policies for object storage:

list

Lists storage policies for object storage.

--policy-name *PolicyName*

Lists details of the specified storage policy, if it exists.

--policy-function *PolicyFunction*

Lists details of the storage policies with the specified function, if any exist.

--verbose

Lists the functions enabled for the storage policies.

create

Creates a storage policy for object storage. The associated configuration files are updated and the ring files are created for the storage policy. The CES monitoring framework distributes the changes to the protocol nodes and restarts the associated services.

PolicyName

Specifies the name of the storage policy.

The policy name must be unique (case insensitive), without spaces, and it must contain only letters, digits, or a dash.

-f *FilesetName*

Specifies the name of the fileset that must be used or created for the storage policy. An existing fileset can be used provided it is not being used for an existing storage policy.

If no fileset name is specified with the command, the policy name is reused for the fileset with the prefix obj_.

--i *MaxNumInodes*

Specifies the inode limit for the new inode space.

--enable-compression

Enables a compression policy. The Swift policy type is replication. If **--enable-compression** is used, **--compression-schedule** must be specified too and vice-versa.

mmobj

Every object stored within a container that is linked to this storage policy is compressed on a scheduled basis. This occurs as a background process. For object retrieval, no decompression is needed because it occurs automatically in the background.

--compression-schedule: "*MM:HH:dd:ww*"

Specifies the compression schedule if **--enable-compression** is used. Schedule needs to be given in the MM:HH:dd:ww format :

MM = 0-59 minutes

Minute after the hour the job should be executed. The range is 0 to 59.

HH = 0-23

Hour in which the job should be executed. Hours are represented as numbers from 0 to 23.

dd = 1-31

The day of a month on which the job should be executed. Days are represented as numbers from 1 to 31.

ww = 0-7 (0=Sun, 7=Sun)

The days of the week the job should be executed. One or more values can be specified (comma separated). Days are represented as numbers from 0 to 7. 0 and 7 represent Sunday. All days of a week are represented by *. Optional. Default is 0.

- Use * for specifying every instance of a unit. For example, dd = * means that the job is scheduled to run every day.
- Comma separated lists are allowed. For example, dd = 1,3,5 means that the job is scheduled to run on every 1st, 3rd, 5th of a month.
- If ww and dd both are specified, the union is used.
- Specifying a range using - is not supported.
- Empty values are allowed for dd and ww. If empty, dd and or ww are not considered.
- Empty values for mm and hh are treated as *.

--enable-file-access

Enables a file-access policy. The file-access policies only exist in the region in which they were created. They do not support the multi-region capability. Objects stored within a container that is linked to this storage policy can be enabled for file protocol access.

Note: The enabled functions are displayed in the functions column of the **mmobj policy list** command output as follows:

- **--enable-compression** compression
- **--enable-file-access** file-and-object-access

change

Changes the state of the specified storage policy.

PolicyName

Specifies the name of the storage policy that needs to be changed.

--default

Sets the specified storage policy to be the default policy.

Note: You cannot set a deprecated storage policy as the default storage policy.

--deprecate *State*

Deprecates the specified storage policy. *State* can be either yes or no.

yes

Sets the state of the specified storage policy as deprecated.

no Sets the state of the specified storage policy as not deprecated.

Note: You cannot deprecate the default storage policy.

--add-local-region

In a multi-region environment, adds the region of the current cluster to the specified storage policy. The associated fileset previously defined for the storage policy must already exist or else it is created.

After the region is added, the multi-region configuration needs to be synced with the other regions by using the **mmobj multiregion** command.

Note: By default, a storage policy only stores objects in the region on which it was created. If the cluster is defined as multi-region, a storage policy can also be made multi-region by adding additional regions to its definition.

--remove-region-number *RegionNumber*

In a multi-region environment, removes a region from the specified storage policy. The associated fileset for the storage policy is not modified.

After the region is removed, the multi-region configuration needs to be synced with the other regions by using the **mmobj multiregion** command.

file-access

Enables file(s) for object access (objectizes) in a unified file and object access environment:

--object-path *ObjectPath*

The fully qualified path of a file or a directory for which you want to enable access through the object interface. If a fully qualified path to a directory is specified then the command enables all the files from that directory for access through the object interface. This is a mandatory parameter for the **mmobj file-access** command if the **--storage-policy** parameter is not specified.

--storage-policy *PolicyName*

The name of the storage policy for which you want to enable files for the object interface. This is a mandatory parameter for the **mmobj file-access** command if the **--object-path** parameter is not specified. If only this parameter is specified, the command enables all files for object interface from the fileset associated with the specified storage policy.

--account-name *AccountName*

The account name for which you want to enable files for access through the object interface. The **--storage-policy** parameter is mandatory if you are using this parameter.

--container-name *ContainerName*

The container name for which you want to enable files for access through the object interface. You must specify the **--storage-policy** and the **--account-name** with this parameter.

--object-name *ObjectName*

The object name for which you want to enable files for access through the object interface. You must specify **--storage-policy**, **--account-name**, and **--container-name** parameters with this parameter.

--node *NodeName*

The node on which the command is to be executed. Optional. If this parameter is not specified, the command is executed from the current node if it is a protocol node. If the current node is not a protocol node, an available protocol node is selected.

multiregion

Administers multi-region object deployment. For more information on multi-region object deployment and capabilities, see *Overview of multi-region object deployment* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

list

Lists the information about the region.

mmobj

enable

Enable the cluster for multi-region support.

Only the first cluster of the region can run the enable command. Subsequent regions join the multi-region cluster during installation with the use of the **--join-region-file** flag of the **mmobj swift base** command.

export

Exports the multi-region configuration environment so that other regions can be installed into the multi-region cluster or other regions can be synced to this region.

If successful, a region checksum is printed in the output. This checksum can be used to ensure different regions are in sync when the **mmobj multiregion import** command is run.

Note: When region-related information changes, such as CES IPs and storage policies, all regions must be updated with the changes.

--region-file *RegionFile*

Specifies a path to store the multi-region data.

This file is created.

import

Imports the specified multi-region configuration environment into this region.

If successful, a region checksum for this region is printed in the output. If the local region configuration matches the imported configuration, the checksums match. If they differ, then it means that some configuration information in the local region needs to be synced to the other regions. This can happen when a configuration change in the local region, such as adding CES IPs or storage policies, has not yet been synced with the other regions. If this is the case, the multi-region configuration for the local region needs to be exported and synced to the other regions.

--region-file *RegionFile*

Specifies the path to a multi-region data file created by using the **mmobj multiregion export** command.

remove

Completely removes a region from the multi-region environment. The removed region will no longer be accessible by other regions.

After the region is removed, the remaining regions need to have their multi-region information synced with this change by using the **mmobj multiregion export** and **mmobj multiregion import** commands.

--region-number *RegionNumber*

Specifies the region number that you need to remove from the multi-region configuration.

--force

Indicates that all the configuration information for the specified region needs to be permanently deleted.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmobj** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To specify configuration of Object protocol service with local keystone and S3 enabled, issue this command:

```
mmobj swift base -g /gpfs/ObjectFS --cluster-hostname cluster-ces-ip.ibm --local-keystone
--enable-s3 --admin-password Passw0rd
```

The system displays output similar to this:

```
mmobj swift base: Creating fileset /dev/ObjectFS object_fileset
mmobj swift base: Configuring Keystone server in /gpfs/ObjectFS/ces/object/keystone
mmobj swift base: Configuring Swift services
Configuration complete
```

2. To list object configuration settings for proxy-server.conf, DEFAULT section, issue this command:

```
mmobj config list --ccrfile proxy-server.conf --section DEFAULT
```

The system displays output similar to this:

```
[DEFAULT]
bind_port = 8080
workers = auto
user = swift
log_level = ERROR
```

3. To change the number of worker processes that each object server launches, update the object-server.conf file as shown here:

```
mmobj config change --ccrfile object-server.conf --section DEFAULT --property workers --value 16
```

4. To change the configuration value of paste.filter_factory which is in the filter:s3_extension section of the keystone-paste.ini configuration file, issue the command:

```
mmobj config change --ccrfile keystone-paste.ini --section filter:s3_extension
--property paste.filter_factory --value keystone.contrib.s3:S3Extension.factory
```

5. To create a new storage policy CompressionTest with the expiration function enabled and with the expiration time specified, issue this command:

```
mmobj policy create CompressionTest --enable-compression
```

The system displays output similar to this:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_CompressionTest
[I] Creating new unique index and build the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

6. To list storage policies for object storage with details of functions available with those storage policies, issue this command:

```
mmobj policy list --verbose
```

The system displays output similar to this:

Index	Name	Deprecated	Fileset	Fileset Path	Functions	Function Details
0	SwiftDefault		object_fileset	/ibm/cesSharedRoot/object_fileset		
11751509160	sof-policy		obj_sof-policy	/ibm/cesSharedRoot/obj_sof-policy	file-and-object-access	regions="1"
11751509230	mysofpolicy		obj_mysofpolicy	/ibm/cesSharedRoot/obj_mysofpolicy	file-and-object-access	regions="1"
11751510260	Test19		obj_Test19	/ibm/cesSharedRoot/obj_Test19		regions="1"

mmobj

7. To enable object access for an account, issue this command:

```
mmobj file-access --storage-policy sof_policy --account-name admin
```

The system displays output similar to the following:

```
Loading objectization configuration from CCR
Fetching storage policy details
Creating container to database map
Performing objectization
Objectization complete
```

8. To enable object access for a container, issue this command:

```
mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1
```

9. To enable object access on a file while specifying a storage policy, issue this command:

```
mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1 --object-name file1.txt
```

10. To enable object access on a file, issue this command:

```
mmobj file-access --object-path /ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_12345/container1/file1.txt
```

11. To list the information about a region, issue this command:

```
mmobj multiregion list
```

The system displays output similar to this:

Region	Endpoint	Cluster Name	Cluster Id
1	RegionOne	europe.gpfs.net	3694106483743716196
2	Region2	asia.gpfs.net	1860802811592373112

12. To set up the initial multi-region environment on the first region, issue this command:

```
mmobj multiregion enable
```

The system displays output similar to this:

```
mmobj multiregion: Multi-region support is enabled in this cluster. Region number: 1
```

13. To export multi-region data for use by other clusters to join multi-region, issue this command:

```
mmobj multiregion export --region-file /tmp/region2.dat
```

The system displays output similar to this:

```
mmobj multiregion: The Object multi-region export file was successfully created: /tmp/region2.dat
mmobj multiregion: Region checksum is: 34632-44791
```

14. To import the specified multi-region configuration environment created by the export command into a region, issue this command:

```
mmobj multiregion import --region-file /tmp/region2.dat
```

The system displays output similar to this:

```
mmobj multiregion: The region config has been updated.
mmobj multiregion: Region checksum is: 34632-44791
```

15. To remove a region designated by region number 2 from a multi-region environment and to remove all configuration information of the specified region, issue this command:

```
mmobj multiregion remove --remove-region-number 2 --force
```

The system displays output similar to this:

```
mmobj multiregion: Permanently removing region 2 (asia.gpfs.net 1860802811592373112) from multi-region configuration.
mmobj multiregion: Updating ring files.
mmobj multiregion: Successfully removed region 2.
Object services on region 2 will need to be unconfigured and its endpoint removed from Keystone.
```

See also

- “mmces command” on page 304
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570

- “mmsmb command” on page 663
- “mmuserauth command” on page 690

Location

/usr/lpp/mmfs/bin

mmperfmon

mmperfmon command

Configures the Performance Monitoring tool and lists the performance metrics.

Synopsis

```
mmperfmon config generate --collectors CollectorNode[,CollectorNode...]
[ --config-file InputFile ]
```

or

```
mmperfmon config add --sensors SensorFile
```

or

```
mmperfmon config update { [--collectors CollectorNode[,CollectorNode...] ]
[ --config-file InputFile ] [ Attribute=value ... ] }
```

or

```
mmperfmon config delete {--all | --sensors Sensor[,Sensor...] }
```

or

```
mmperfmon config show [--config-file OutputFile]
```

or

```
mmperfmon query Metric[,Metric...] | Key[,Key...] | NamedQuery
[StartTime EndTime | Duration]
[Options]
```

or

```
mmperfmon query compareNodes ComparisonMetric
[StartTime EndTime | Duration]
[Options]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

Description

mmperfmon config modifies the Performance Monitoring tool by updating the configuration stored in IBM Spectrum Scale. It can be used to generate an initial configuration, to update reporting periods of different sensors, or to restrict sensors to a given set of nodes.

mmperfmon query is used to query metrics in a cluster from the performance metrics collector. Output can be delivered either in a raw format, formatted table layout or as a CSV export.

In addition to metrics known by the performance collector, the **mmperfmon query** command can also run predefined named queries or use predefined computed metrics. You can specify a bucket size for each record to return in number of seconds and the number of buckets to retrieve. You can also specify the duration or a time range for which the query can run.

Parameters

config

generate

Generates the configuration of the Performance Monitoring tool.

Note: Once the configuration has been generated, do not forget to turn on monitoring through the **mmchnode** command.

--collectors *CollectorNode[,CollectorNode...]* specifies the set of collectors to which the sensors report their performance measurements. The number of collectors that each sensor shall report to may be specified through **colRedundancy** parameter in the template sensor configuration file (see **--config-file**).

--config-file *InputFile* specifies the template sensor configuration file to use. If this option is not provided the `/opt/IBM/zimon/defaults/ZIMonSensors.cfg` file is used.

add

Adds a new sensor to the Performance Monitoring tool.

--sensors *SensorFile* adds the sensors specified in *SensorFile* to the sensor configuration. Multiple sensors in the configuration file need to be separated by a comma. Following is a sample *SensorFile*:

```
sensors = {
name = "MySensor"
# sensor disabled by default
period = 0
type = "Generic"
}
```

The generic sensor and a sensor-specific configuration file need to be installed on all the nodes where the generic sensor is to be activated.

update

Updates the existing configuration.

--collectors *CollectorNode[,CollectorNode...]* updates the collectors to be used by the sensors (see **config generate** for details).

--config-file *InputFile* specifies a template sensor configuration file to use. This overwrites the currently used configuration with the configuration specified in *InputFile*.

Attribute=value ... specifies a list of attribute value assignments. This sets the value of attribute *Attribute* to *value*.

delete

Removes configuration of the Performance Monitoring tool or the specified sensors.

--sensors *Sensor[,Sensor...]* removes the sensors with the specified names from the performance monitoring configuration.

--all removes the entire performance monitoring configuration from IBM Spectrum Scale.

show

Displays the currently active performance monitoring configuration. Specifies the following options:

--config-file *OutputFile* specifies that the output will be saved to the *OutputFile*. This option is optional.

query

Metric[,Metric...] specifies a comma separated list of metrics for displaying in the output.

Key[,Key...] specifies a key that can consist of a node name, sensor group, or optional additional filters, and metrics that are separated by the pipe symbol (`|`). For example:

```
"cluster1.ibm.com|CTDBStats|locking|db_hop_count_bucket_00"
```

mmperfmon

NamedQuery specifies the name of a predefined query.

compareNodes compares the specified metrics for all nodes in the system. The query creates one column per existing node and only one metric can be compared.

ComparisonMetric specifies the name of the metric to be compared when using the **compareNodes** query.

StartTime specifies the start timestamp for query in the YYYY-MM-DD-hh:mm:ss format.

EndTime specifies the end timestamp for query in the YYYY-MM-DD-hh:mm:ss format. If it is not specified, the query will return results until the present time.

Duration specifies the number of seconds into the past from present time or *EndTime*.

Options specifies the following options:

- **-N** or **--Node NODENAME** specifies the node from where the metrics should be retrieved.
For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.
- **--bucket_size BUCKET_SIZE** specifies the bucket size (number of seconds), default is 1.
- **--number_buckets NUMBER_BUCKETS** specifies the number of buckets (records) to display, default is 10.
- **--filter FILTER** specifies the filter criteria for the query to run. To see the list of filters in the node use the **mmperfmon query --list filters** command.
- **--format FORMAT** specifies a common format for all columns.
- **--csv** provides the output in the CSV format.
- **--raw** provides the output in a raw format rather than a tabular format.
- **--short** displays the column names in a short form when there are too many to fit into a row.
- **--nice** displays the column headers in the output in a bold and underlined typeface.
- **--resolve** displays the resolved computed metrics and metrics that are used.
- **--list {computed | metrics | keys | filters | queries | all}** lists the following information:
 - **computed** displays the computed metrics.
 - **metrics** displays the metrics.
 - **keys** lists the keys.
 - **filters** lists the filters.
 - **queries** lists the available predefined queries.
 - **all** displays the computed metrics, metrics, keys, filters, and queries.

Exit status

- | | |
|---|--|
| 0 | Successful completion. |
| 1 | Invalid arguments given |
| 2 | Invalid option |
| 3 | No node found with a running performance collector |
| 4 | Performance collector backend signaled bad query, for example, no data for this query. |

Security

You must have root authority to run the **mmperfmon** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For

more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To generate configuration for the c89f8v03 collector node, issue the command:

```
mmperfmon config generate --collectors c89f8v03
```

The system displays output similar to this:

```
mmperfmon: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
Tue Oct 27 20:40:07 EDT 2015: mmcommon pushSdr_async: mmsdrfs propagation started
```

2. To add /tmp/SensorFile sensor to the Performance Monitoring tool issue the command:

```
mmperfmon config add --sensors /tmp/SensorFile
```

The system displays output similar to this:

```
mmperfmon: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
Tue Oct 27 20:44:33 EDT 2015: mmcommon pushSdr_async: mmsdrfs propagation started
# mmperfmon config show | tail -12
```

```
{
    name = "NFSIO"
    period = 0
    proxyCmd = "/opt/IBM/zimon/GaneshaProxy"
    restrict = "cesNodes"
    type = "Generic"
},
{
    name = "TestAdd"
    period = 4
}
smbstat = ""
```

3. To update the NFSIO.period value to 5, issue the command:

```
# mmperfmon config update NFSIO.period=5
```

The system displays output similar to this:

```
mmperfmon: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
Tue Oct 27 20:47:53 EDT 2015: mmcommon pushSdr_async: mmsdrfs propagation started
```

```
# mmperfmon config show | tail -9
},
{
    name = "NFSIO"
    period = 5
    proxyCmd = "/opt/IBM/zimon/GaneshaProxy"
    restrict = "cesNodes"
    type = "Generic"
}
smbstat = ""
```

4. To remove the TestAdd sensor, issue the following command:

```
# mmperfmon config delete --sensors TestAdd
```

The system displays output similar to this:

```
mmperfmon: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
Tue Oct 27 20:46:23 EDT 2015: mmcommon pushSdr_async: mmsdrfs propagation started
Tue Oct 27 20:46:28 EDT 2015: mmcommon pushSdr_async: mmsdrfs propagation completed; mmdsh rc=0
```

```
# mmperfmon config show | tail -12
{
```

mmpmmon

```
        name = "GPFSDiskCap"
        period = 0
    },
    {
        name = "NFSIO"
        period = 0
        proxyCmd = "/opt/IBM/zimon/GaneshProxy"
        restrict = "cesNodes"
        type = "Generic"
    }
}
smbstat = ""
```

5. To display the currently active performance monitoring configuration, issue the command:

```
# mmpmmon config show
```

The system displays output similar to this:

```
cephMon = "/opt/IBM/zimon/CephMonProxy"
cephRados = "/opt/IBM/zimon/CephRadosProxy"
colCandidates = "c89f8v03"
colRedundancy = 1
collectors = {
    host = ""
    port = "4739"
}
config = "/opt/IBM/zimon/ZIMonSensors.cfg"
ctdbstat = ""
daemonize = T
hostname = ""
ipfixinterface = "0.0.0.0"
logfile = "/var/log/zimon/ZIMonSensors.log"
loglevel = "info"
mmcmd = "/opt/IBM/zimon/MMCmdProxy"
mmdfcmd = "/opt/IBM/zimon/MMDFProxy"
mmpmon = "/opt/IBM/zimon/MmpmonSockProxy"
piddir = "/var/run"
release = "4.2.0-0"
sensors = {
    name = "CPU"
    period = 1
},
{
    name = "Load"
    period = 1
},
{
    name = "Memory"
    period = 1
},
{
    name = "Network"
    period = 1
},
{
    name = "Netstat"
    period = 0
},
{
    name = "Diskstat"
    period = 0
},
{
    name = "DiskFree"
    period = 600
},
{
    name = "GPFSDisk"
```

```

    period = 0
  },
  {
    name = "GPFSFilesystem"
    period = 1
  },
  {
    name = "GPFSNSDDisk"
    period = 1
    restrict = "nsdNodes"
  },
  {
    name = "GPFSPoolIO"
    period = 0
  },
  {
    name = "GPFSVFS"
    period = 1
  },
  {
    name = "GPFSIOC"
    period = 0
  },
  {
    name = "GPFSVIO"
    period = 0
  },
  {
    name = "GPFSPPDisk"
    period = 1
    restrict = "nsdNodes"
  },
  {
    name = "GPFSvFLUSH"
    period = 0
  },
  {
    name = "GPFSNode"
    period = 1
  },
  {
    name = "GPFSNodeAPI"
    period = 1
  },
  {
    name = "GPFSFilesystemAPI"
    period = 1
  },
  {
    name = "GPFSLROC"
    period = 0
  },
  {
    name = "GPFSCHMS"
    period = 0
  },
  {
    name = "GPFSAFM"
    period = 0
  },
  {
    name = "GPFSAFMFS"
    period = 0
  },
  {
    name = "GPFSAFMFSET"
    period = 0
  }

```

mmpfmon

```
},
{
    name = "GPFSRPCS"
    period = 0
},
{
    name = "GPFSFilesetQuota"
    period = 3600
},
{
    name = "GPFSDiskCap"
    period = 0
},
{
    name = "NFSIO"
    period = 0
    proxyCmd = "/opt/IBM/zimon/GaneshaProxy"
    restrict = "cesNodes"
    type = "Generic"
}
smbstat = ""
```

6. To list metrics by key, for given node, sensor group and metrics, issue this command:

```
mmpfmon query "cluster1.ibm.com|CTDBDBStats|locking|db_hop_count_bucket_00"
```

The system displays output similar to this:

Row	Timestamp	db_hop_count_bucket_00
1	2015-04-08-12:54:53	0
2	2015-04-08-12:54:54	0
3	2015-04-08-12:54:55	0
4	2015-04-08-12:54:56	0
5	2015-04-08-12:54:57	0
6	2015-04-08-12:54:58	0
7	2015-04-08-12:54:59	0
8	2015-04-08-12:55:00	0
9	2015-04-08-12:55:01	0
10	2015-04-08-12:55:02	0

7. To list the two metrics `nfs_read_lat` and `nfs_write_lat` for a specific time range, filtered by an export and NFS version with 60-seconds-buckets (one record represents 60 seconds), issue this command:

```
mmpfmon query nfs_read_lat,nfs_write_lat 2014-12-19-11:15:00 2014-12-19-11:20:00 --filter export=/ibm/gpfs/nfsexport,nfs_ver=NFSv3 -b 60
```

The system displays output similar to this:

Row	Timestamp	nfs_read_lat	nfs_write_lat
1	2015-04-10-09:24:00	0	0
2	2015-04-10-09:24:10	0	0
3	2015-04-10-09:24:20	0	0
4	2015-04-10-09:24:30	0	0
5	2015-04-10-09:24:40	0	0
6	2015-04-10-09:24:50	0	0
7	2015-04-10-09:25:00	0	0
8	2015-04-10-09:25:10	0	0
9	2015-04-10-09:25:20	0	0
10	2015-04-10-09:25:30	0	0
11	2015-04-10-09:25:40	0	0
12	2015-04-10-09:25:50	45025738	1882453623
13	2015-04-10-09:26:00	0	0
14	2015-04-10-09:26:10	0	0
15	2015-04-10-09:26:20	0	0
16	2015-04-10-09:26:30	0	0
17	2015-04-10-09:26:40	0	0
18	2015-04-10-09:26:50	0	0

8. To list all available filters, issue this command:

```
mmpfmon query --list filters
```

The system displays output similar to this:

Available Filters:

```

node
    gpfs-21.localnet.com
    gpfs-22.localnet.com
protocol
    smb2
db_name
    account_policy
    autorid
    brlock
    ctdb
    dbwrap_watchers
    g_lock
    group_mapping
    leases
    locking
    netlogon_creds_cli
    notify_index
    passwd
    registry
    secrets
    serverid
    share_info
    smbXsrv_open_global
    smbXsrv_session_global
    smbXsrv_tcon_global
    smbXsrv_version_global
gpfs_fs_name
    fs0
    gpfs0
gpfs_cluster_name
    gpfs-cluster-2.localnet.com
mountPoint
    /
    /boot
    /dev
    /dev/shm
    /gpfs/fs0
    /mnt/gpfs0
    /run
    /sys/fs/cgroup
operation
    break
    cancel
    close
    create
    find
    flush
    getinfo
    ioctl
    keepalive
    lock
    logoff
    negprot
    notify
    read
    sesssetup
    setinfo
    tcon
    tdis
    write
sensor
    CPU
    CTDBDBStats
    CTDBStats

```

mmperfmon

```
DiskFree
GPFSFilesystemAPI
GPFSVFS
Load
Memory
Network
SMBGlobalStats
SMBStats
netdev_name
eth0
lo
```

9. To run a named query for export /ibm/gpfs/nfsexport and **nfs_ver NFSv3**, using default bucket size of 1 second, showing last 10 buckets , issue this command:

```
mmperfmon query nfsIOrate --filter export=/ibm/gpfs/nfsexport,nfs_ver=NFSv3,node=cluster1.ibm.com
```

The system displays output similar to this:

Legend:

```
1: cluster1.ibm.com|NFSIO|/ibm/gpfs/nfsexport|NFSv3|nfs_read_ops
2: cluster2.ibm.com|NFSIO|/ibm/gpfs/nfsexport|NFSv3|nfs_write_ops
```

Row	Timestamp	nfs_read_ops	nfs_write_ops
1	2015-05-11-13:32:57	0	0
2	2015-05-11-13:32:58	90	90
3	2015-05-11-13:32:59	90	90
4	2015-05-11-13:33:00	90	91
5	2015-05-11-13:33:01	91	90
6	2015-05-11-13:33:02	91	92
7	2015-05-11-13:33:03	89	88
8	2015-05-11-13:33:04	91	92
9	2015-05-11-13:33:05	93	92
10	2015-05-11-13:33:06	89	89

10. To run a named query for export /ibm/gpfs/nfsexport and **nfs_ver NFSv3**, using bucket size of 1 minute, showing last 20 buckets (= 20 minutes), issue this command:

```
mmperfmon query nfsIOrate --filter export=/ibm/gpfs/nfsexport,nfs_ver=NFSv3,node=cluster1.ibm.com -n 20 -b 60
```

The system displays output similar to this:

Legend:

```
1: cluster1.ibm.com|NFSIO|/ibm/gpfs/nfsexport|NFSv3|nfs_read_ops
2: cluster2.ibm.com|NFSIO|/ibm/gpfs/nfsexport|NFSv3|nfs_write_ops
```

Row	Timestamp	nfs_read_ops	nfs_write_ops
1	2015-05-11-13:31:00	0	0
2	2015-05-11-13:32:00	280	280
3	2015-05-11-13:33:00	820	820
4	2015-05-11-13:34:00	0	0
5	2015-05-11-13:35:00	0	0
6	2015-05-11-13:36:00	0	0
7	2015-05-11-13:37:00	0	0
8	2015-05-11-13:38:00	0	0
9	2015-05-11-13:39:00	1000	1000
10	2015-05-11-13:40:00	1000	1000
11	2015-05-11-13:41:00	0	0
12	2015-05-11-13:42:00	0	0
13	2015-05-11-13:43:00	0	0
14	2015-05-11-13:44:00	2000	2000
15	2015-05-11-13:45:00	0	0
16	2015-05-11-13:46:00	0	0
17	2015-05-11-13:47:00	1000	1000
18	2015-05-11-13:48:00	1000	1000
19	2015-05-11-13:49:00	0	0
20	2015-05-11-13:50:00	0	0

11. To run a **compareNodes** query for the *cpu_user* metric, issue this command:

```
mmpfmon query compareNodes cpu_user
```

The system displays output similar to this:

Legend:

```
1: cluster1.ibm.com|CPU|cpu_user
2: cluster2.ibm.com|CPU|cpu_user
```

Row	Timestamp	cluster1	cluster2
1	2015-05-11-13:53:54	0.5	0.25
2	2015-05-11-13:53:55	0.5	0.25
3	2015-05-11-13:53:56	0.25	0.25
4	2015-05-11-13:53:57	0.5	0.25
5	2015-05-11-13:53:58	0.25	0.75
6	2015-05-11-13:53:59	0.5	0.25
7	2015-05-11-13:54:00	0.25	0.25
8	2015-05-11-13:54:01	0.5	0.25
9	2015-05-11-13:54:02	0.25	0.25
10	2015-05-11-13:54:03	0.5	0.25

See also

- “`mmdumpperfdata` command”

Location

```
/usr/lpp/mmfs/bin
```

mmpmon

mmpmon command

Manages performance monitoring and displays performance information.

Synopsis

```
mmpmon [-i CommandFile] [-d IntegerDelayValue] [-p]
        [-r IntegerRepeatValue] [-s] [-t IntegerTimeoutValue]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Before you attempt to use **mmpmon**, it is a good idea to review this command entry, then read the entire topic *Monitoring GPFS I/O performance with the mmpmon command* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Use the **mmpmon** command to manage GPFS performance monitoring functions and display performance monitoring data. The **mmpmon** command reads requests from an input file or standard input (stdin), and writes responses to standard output (stdout). Error messages go to standard error (stderr). Prompts, if not suppressed, go to stderr.

You can run **mmpmon** so that it continually reads input from a pipe. That is, the driving script or application never sends an end of file. In this scenario, it is a good idea to set the **-r** option to 1, or to use the default value of 1. This setting prevents the command from caching input records. In doing so it avoids unnecessary memory consumption.

This command cannot be run from a Windows node.

Results

The performance monitoring request is sent to the GPFS daemon that is running on the same node that is running the **mmpmon** command.

All results from the request are written to stdout.

The command has two output formats:

- Human readable, intended for direct viewing.
In this format, the results are keywords that describe the value presented, followed by the value. Here is an example:
disks: 2
- Machine readable, an easily parsed format intended for further analysis by scripts or applications.
In this format, the results are strings with values presented as keyword/value pairs. The keywords are delimited by underscores (_) and blanks to make them easier to locate.

For details on how to interpret the **mmpmon** command results, see the topic *Monitoring GPFS I/O performance with the mmpmon command* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

-i *CommandFile*

The input file contains **mmpmon** command requests, one per line. Use of the **-i** flag implies use of

the **-s** flag. For interactive use, omit the **-i** flag. In this case, the input is then read from stdin, allowing **mmpmon** to take keyboard input or output piped from a user script or application program.

Leading blanks in the input file are ignored. A line beginning with a number sign (#) is treated as a comment. Leading blanks in a line whose first non-blank character is a number sign (#) are ignored.

The input requests to the **mmpmon** command are as follows:

fs_io_s

Displays I/O statistics per mounted file system.

io_s

Displays I/O statistics for the entire node.

nlist add name [name...]

Adds node names to a list of nodes for **mmpmon** processing.

nlist del

Deletes a node list.

nlist new name [name...]

Creates a node list.

nlist s

Shows the contents of the current node list.

nlist sub name [name...]

Deletes node names from a list of nodes for **mmpmon** processing.

once request

Indicates that the request is to be performed only once.

qosio

Displays statistics for Quality of Service for I/O operations (QoS).

reset

Resets statistics to zero.

rhist nr

Changes the request histogram facility request size and latency ranges.

rhist off

Disables the request histogram facility. This value is the default.

rhist on

Enables the request histogram facility.

rhist p

Displays the request histogram facility pattern.

rhist reset

Resets the request histogram facility data to zero.

rhist s

Displays the request histogram facility statistics values.

rpc_s

Displays the aggregation of execution time for remote procedure calls (RPCs).

rpc_s size

Displays the RPC execution time according to the size of messages.

ver

Displays **mmpmon** version.

mmpmon

vio_s [**f** **rg** *RecoveryGroupName* [**da** *DeclusteredArray* [**v** *Vdisk*]]] [**reset**]

Displays GPFS Native RAID vdisk I/O statistics. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

vio_s_reset [**f** **rg** *RecoveryGroupName* [**da** *DeclusteredArray* [**v** *Vdisk*]]]

Resets GPFS Native RAID vdisk I/O statistics. For more information about GPFS Native RAID, see *IBM Spectrum Scale RAID: Administration*.

Options

-d *IntegerDelayValue*

Specifies a number of milliseconds to sleep after one invocation of all the requests in the input file. The default value is 1000. This value must be an integer greater than or equal to 500 and less than or equal to 8000000.

The command processes the input file in the following way:

1. The command processes each request in the input file sequentially. It reads a request, processes it, sends it to the GPFS daemon, and waits for the response. When it receives the response, the command processes it and displays the results of the request. The command then goes on to the next request in the input file.
2. When the command processes all the requests in the input file, it sleeps for the specified number of milliseconds.
3. When the command wakes, it begins another cycle of processing, beginning with Step 1. The number of repetitions depends on the value of the **-r** flag.

-p Indicates to generate output that can be parsed by a script or program. If this option is not specified, human-readable output is produced.

-r *IntegerRepeatValue*

Specifies the number of times to run all the requests in the input file.

The default value is one. Specify an integer between zero and 8000000. Zero means to run forever, in which case processing continues until it is interrupted. This feature is used, for example, by a driving script or application program that repeatedly reads the result from a pipe.

The **once** prefix directive can be used to override the **-r** flag. See the description of **once** in the topic *Monitoring GPFS I/O performance with the mmpmon command* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-s Indicates to suppress the prompt on input.

Use of the **-i** flag implies use of the **-s** flag. For use in a pipe or with redirected input (<), the **-s** flag is preferred. If not suppressed, the prompts go to standard error (stderr).

-t *IntegerTimeoutValue*

Specifies a number of seconds that the command waits for responses from the GPFS daemon before it fails.

The default value is 60. This value must be an integer greater than or equal to 1 and less than or equal to 8000000.

Exit status

- | | |
|---|--|
| 0 | Successful completion. |
| 1 | Various errors, including insufficient memory, input file not found, incorrect option, and others. |
| 3 | Either no commands were entered interactively, or the input file did not contain any mmpmon commands. The input file was empty, or consisted of all blanks or comments. |
| 4 | mmpmon terminated due to a request that was not valid. |
| 5 | An internal error occurred. |

111 An internal error occurred. A message follows.

Restrictions

1. Up to five instances of **mmpmon** can be run on a node concurrently. However, concurrent users might interfere with each other. See the topic *Monitoring GPFS I/O performance with the mmpmon command* in the *IBM Spectrum Scale: Administration and Programming Reference*.
2. Do not alter the input file while **mmpmon** is running.
3. The input file must contain valid input requests, one per line. When **mmpmon** finds an invalid request, it issues an error message and terminates. The command processes input requests that appear in the input file before the first invalid request.

Security

The **mmpmon** command must be run by a user with root authority, on the node for which you want statistics.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. Assume that **infile** contains these requests:

```
ver
io_s
fs_io_s
rhist off
```

The following command is issued:

```
mmpmon -i infile -r 10 -d 5000
```

The output (sent to stdout) is similar to the following example output:

```
mmpmon node 192.168.1.8 name node1 version 3.1.0
mmpmon node 192.168.1.8 name node1 io_s OK
timestamp:      1083350358/935524
bytes read:      0
bytes written:   0
opens:           0
closes:          0
reads:           0
writes:          0
readdir:         0
inode updates:   0
mmpmon node 192.168.1.8 name node1 fs_io_s status 1
no file systems mounted
mmpmon node 192.168.1.8 name node1 rhist off OK
```

The requests in the input file are run 10 times, with a delay of 5000 milliseconds (5 seconds) between invocations.

2. This example uses the same parameters as the previous example, but with the **-p** flag:

```
mmpmon -i infile -p -r 10 -d 5000
```

The output (sent to stdout) is similar to the following example output:

```
_ver_ _n_ 192.168.1.8 _nn_ node1 _v_ 2 _lv_ 3 _vt_ 0
_io_s_ _n_ 192.168.1.8 _nn_ node1 _rc_ 0 _t_ 1084195701 _tu_ 350714 _br_ 0 _bw_ 0 _oc_ 0
_cc_ 0 _rdc_ 0 _wc_ 0 _dir_ 0 _iu_ 0
_fs_io_s_ _n_ 192.168.1.8 _nn_ node1 _rc_ 1 _t_ 1084195701 _tu_ 364489 _cl_ - _fs_ - _rhist_
_n_ 192.168.1.8 _nn_ node1 _req_ off _rc_ 0 _t_ 1084195701 _tu_ 378217
```

mmpmon

3. This example uses the **fs_io_s** option with a mounted file system:

```
mmpmon node 198.168.1.8 name node1 fs_io_s OK
cluster: node1.localdomain
filesystem: gpfs1
disks: 1
timestamp: 1093352136/799285
bytes read: 52428800
bytes written: 87031808
opens: 6
closes: 4
reads: 51
writes: 83
readdir: 0
inode updates: 11
```

```
mmpmon node 198.168.1.8 name node1 fs_io_s OK
cluster: node1.localdomain
filesystem: gpfs2
disks: 2
timestamp: 1093352136/799285
bytes read: 87031808
bytes written: 52428800
opens: 4
closes: 3
reads: 12834
writes: 50
readdir: 0
inode updates: 9
```

4. This example is the same as the previous one, but with the **-p** flag:

```
_fs_io_s_n_198.168.1.8_nn_node1_rc_0_t_1093352061_tu_93867_cl_node1.localdomain
_fs_gpfs1_d_1_br_52428800_bw_87031808_oc_6_cc_4_rdc_51_wc_83_dir_0_iu_10
_fs_io_s_n_198.168.1.8_nn_node1_rc_0_t_1093352061_tu_93867_cl_node1.localdomain
_fs_gpfs2_d_2_br_87031808_bw_52428800_oc_4_cc_3_rdc_12834_wc_50_dir_0_iu_8
```

This output consists of two strings.

5. This example uses the **io_s** option with a mounted file system:

```
mmpmon node 198.168.1.8 name node1 io_s OK
timestamp: 1093351951/587570
bytes read: 139460608
bytes written: 139460608
opens: 10
closes: 7
reads: 12885
writes: 133
readdir: 0
inode updates: 14
```

6. This example is the same as the previous one, but with the **-p** flag:

```
_io_s_n_198.168.1.8_nn_node1_rc_0_t_1093351982_tu_356420_br_139460608
_bw_139460608_oc_10_cc_7_rdc_0_wc_133_dir_0_iu_14
```

This output consists of one string.

Location

/usr/lpp/mmfs/bin

mmprotocoltrace command

Starts, stops, and monitors tracing for the CES protocols.

Synopsis

```
mmprotocoltrace start <identifier> [<identifier>...] [-c <clientIP >]
                    [-d <duration >] [-l <logFileDir >] [-n <nodes >] [-f]
```

or

```
mmprotocoltrace stop <identifier> [<identifier>...]
```

or

```
mmprotocoltrace status <identifier> [<identifier>...] [-v]
```

or

```
mmprotocoltrace clear <identifier> [<identifier>...] [-f]
```

or

```
mmprotocoltrace reset <identifier> [<identifier>...]
```

or

```
mmprotocoltrace {config}
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Notice: This command has common function to other existing commands. As such, the function may, at any time in a future release, be rolled into other commands and immediately deprecated from use without prior notice. Information about the change and what commands replace it would be provided in some format at the time of that change. Users should avoid using this command in any type of automation or scripting or be advised a future change may break that automation without prior notice.

Description

Use the **mmprotocoltrace** command to trace SMB, Object, or Networking operations. You can run this command from any node in a cluster. You can start, stop, reset, or display the status of a trace. You can set the duration of a trace to avoid excessive logging. Use the **mmprotocoltrace** command to control tracing for SMB. You can start, stop, reset, check or display the status of a trace with this command. It also controls the timeouts for the traces to avoid excessive logging.

Note: The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

For more information about using this command, see the topic on CES tracing and debug data collection in the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

options

Specifies one of the following trace options:

-d *Duration*

Specifies the trace duration in minutes. The default is **10**.

mmprotocoltrace

-l *LogFileDir*

Specifies the name of the directory that contains the log and tar files that are created by the trace. The directory name cannot be a shared directory. The default is a directory in /tmp/mmfs that is named by the trace type and time.

-N *Nodes*

Specifies a comma-separated list of node names to run trace on. The default is **all**.

-c *ClientIPs*

Specifies a comma-separated list of client IP addresses to trace. The default is **all**. This parameter applies only to SMB traces and Network traces.

-f

Forces an action to occur. Affects only the **clear** command.

-v

Verbose output. Affects only the **status** command.

command

Specifies one of the following trace commands:

start

Starts tracing for the specified component.

stop

Stops tracing for the specified component.

status

Displays the status of the specified component.

config

Displays the current contents of the configuration file.

clear

Clears the trace records from the trace list.

reset

Resets the nodes to the default state that is defined in the configuration file.

identifier

Specifies one of the following components:

smb

Traces the SMB service.

object

Traces the Object service.

network

Traces the Network service.

Exit status

0 Successful completion.

Nonzero

A failure occurred.

Security

You must have root authority to run the **mmprotocoltrace** command.

The node on which the command is run must be able to process remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See

the information about the requirements for administering a GPFS system in the *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. The following command starts an SMB trace on all the CES nodes in the cluster for a duration of 10 minutes (the default duration):

```
mmprotocoltrace -N all start smb
```

The response to the command is similar to the following:

```
Trace '3f36dbed-b567-4566-9beb-63b6420bbb2d' created successfully
```

2. The following command starts an SMB trace on the **mobile1** node for a duration of 10 minutes.

```
mmprotocoltrace -N mobile1 start smb
```

3. The following command starts an SMB trace of two clients for a duration of 20 minutes:

```
mmprotocoltrace -d 20 -c 192.168.0.1,192.168.0.2 start smb
```

4. The following command displays the status of an SMB trace:

```
mmprotocoltrace status smb
```

The status that is displayed is similar to the following:

```
Trace ID:    d11145ea-9e9a-4fb0-ae8d-7cb48e49ecc2
State:      WAITING
User ID:    root
Protocol:   smb
Start Time: 11:11:37 05/05/2015
End Time:   11:21:37 05/05/2015
Client IPs: []
Origin Node: swift-test-08.stglab.manchester.uk.ibm.com
Nodes:
  Node Name:  swift-test-07.stglab.manchester.uk.ibm.com
  State:      WAITING
  Trace Location: /dump/ftdc/smb.20150505_111136.trc
  Pids:       []

  Node Name:  swift-test-08.stglab.manchester.uk.ibm.com
  State:      WAITING
  Trace Location: /dump/ftdc/smb.20150505_111136.trc
  Pids:       []
```

5. The following command stops an SMB trace:

```
mmprotocoltrace stop smb
```

The response to this command is similar to the following:

```
Stopping traces
Trace '01239483-be84-wev9-a2d390i9ow02' stopped for smb
Waiting for traces to complete
Waiting for node 'node1'
Waiting for node 'node2'
Finishing trace '01239483-be84-wev9-a2d390i9ow02'
Trace tar file has been written to '/tmp/mmfs/smb.20150513_162322.trc/smb.trace.20150513_162542.tar.gz'
```

6. The following command clears an SMB trace from the trace file:

```
mmprotocoltrace clear smb
```

The response to this command is similar to the following:

```
Trace for protocol 'smb' cleared successfully.
```

Examples

1. To start an SMB trace, issue this command:

```
mmprotocoltrace start smb
```

mmprotocoltrace

The system displays output similar to this:

```
Trace '3f36dbed-b567-4566-9beb-63b6420bbb2d' created successfully
```

2. To view the status of the SMB trace, issue this command:

```
mmprotocoltrace status smb
```

The system displays output similar to this:

```
Trace ID:    d11145ea-9e9a-4fb0-ae8d-7cb48e49ecc2
State:      WAITING
User ID:    root
Protocol:   smb
Start Time: 11:11:37 05/05/2015
End Time:   11:21:37 05/05/2015
Client IPs: []
Origin Node: swift-test-08.stglab.manchester.uk.ibm.com
Nodes:
  Node Name:  swift-test-07.stglab.manchester.uk.ibm.com
  State:      WAITING
  Trace Location: /dump/ftdc/smb.20150505_111136.trc
  Pids:      []
  Node Name:  swift-test-08.stglab.manchester.uk.ibm.com
  State:      WAITING
  Trace Location: /dump/ftdc/smb.20150505_111136.trc
  Pids:      []
```

3. To stop the SMB trace, issue this command:

```
mmprotocoltrace stop smb
```

The system displays output similar to this:

```
Stopping traces
Trace '01239483-be84-wev9-a2d390i9ow02' stopped for smb
Waiting for traces to complete
Waiting for node 'node1'
Waiting for node 'node2'
Finishing trace '01239483-be84-wev9-a2d390i9ow02'
Trace tar file has been written to '/tmp/mmfs/smb.20150513_162322.trc/smb.trace.20150513_162542.tar.gz'
```

4. To clear the SMB trace from the trace file, issue this command:

```
mmprotocoltrace clear smb
```

The system displays output similar to this:

```
Trace for protocol 'smb' cleared successfully.
```

See also

- “mmaddcallback command” on page 226
- “mmces command” on page 304
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmsmb command” on page 663
- “mmuserauth command” on page 690

Location

```
/usr/lpp/mmfs/bin
```


mmpsnap command

Creates or deletes identical snapshots on the cache and home clusters, or shows the status of snapshots that have been queued up on the gateway nodes.

Synopsis

```
mmpsnap Device create -j FilesetName [{[--comment Comment] [--uid ClusterUID]} | --rpo] [--wait]
```

or

```
mmpsnap Device delete -s SnapshotName -j FilesetName
```

or

```
mmpsnap Device status -j FilesetName
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmpsnap** command creates or deletes identical snapshots on the cache and home clusters, or shows the status of snapshots that have been queued up on the gateway nodes.

Parameters

Device

Specifies the device name of the file system.

create

Creates a fileset level snapshot in cache and a snapshot with the same name at home. The snapshot at home could be fileset level or file system level, depending on whether the exported path is an independent fileset or file system.

-j *FilesetName*

Specifies the name of the fileset.

--comment *Comment*

Optional; specifies user-defined text to be prepended to the snapshot name (thereby customizing the name of the snapshot).

--uid *ClusterUID*

Optional; specifies a unique identifier for the cache site. If not specified, this defaults to the GPFS cluster ID.

--rpo

Optional; specifies that user recovery point objective (RPO) snapshots are to be created for a primary fileset. This option cannot be specified with the **--comment** and **--uid** options.

--wait

Optional; makes the creation of cache and home snapshots a synchronous process. When specified, **mmpsnap** does not return until the snapshot is created on the home cluster. When not specified, **mmpsnap** is asynchronous and returns immediately rather than waiting for the snapshot to be created at home.

delete

Deletes the local and remote copies of the specified snapshot; AFM automatically figures out the remote device and fileset.

-s *SnapshotName*

Specifies the name of the snapshot to be deleted. A snapshot name is constructed as follows:

mmpsnap

```
{commentString}-psnap-{clusterId}-{fsUID}-{fsetID}-{timestamp}
```

Where *timestamp* has the form YY-MM-DD-HH-MM-SS.

In the following example, a comment string was not provided:

```
psnap-14133737607146558608-C0A8AA04:4EDD34DF-1-11-12-05-14-32-10
```

status

Shows the status of snapshots that have been queued up on the gateway nodes. The status includes the following pieces of information:

- Last successful snapshot (obtained through **mmlsfileset --afm**)
- Status of the current snapshot process.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmpsnap** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To create a fileset level snapshot in cache of a single-writer fileset called **sw** in file system **fs1** issue this command:

```
mmpsnap fs1 create -j sw
```

The system displays output similar to the following:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
Snapshot psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 created with id 8.
Snapshot psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 created at the satellite.
Core snapshot has been queued.
```

To display the snapshot issue this command:

```
mmlssnapshot fs1 -j sw
```

The system displays output similar to the following:

```
Snapshots in file system fs1:
Directory SnapId Status Created Fileset
psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 8 Valid Thu Mar 1 02:27:29 2012 sw
```

To show that the snapshot is also created at home, issue this command:

```
mmlssnapshot fs1
```

The system displays output similar to the following:

Snapshots in file system fs1:
Directory SnapId Status Created Fileset
psnap-13882361812785649740-C0A80E85:4F44B305-59-12-03-01-02-27-28 8 Valid Thu Mar 1 02:23:16 2012

See also

- “mmafmconfig command” on page 248
- “mmafmctl command” on page 251
- “mmafmlocal command” on page 264
- “mmchattr command” on page 321
- “mmchconfig command” on page 331
- “mmchfileset command” on page 365
- “mmchfs command” on page 371
- “mmcrfileset command” on page 408
- “mmcrfs command” on page 414
- “mmlsconfig command” on page 526
- “mmlsfileset command” on page 532
- “mmlsfs command” on page 536

Location

/usr/lpp/mmfs/bin

mmputac1

mmputac1 command

Sets the GPFS access control list for the specified file or directory.

Synopsis

```
mmputac1 [-d] [-i InFilename] Filename
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmputac1** command to set the ACL of a file or directory.

If the **-i** option is not used, the command expects the input to be supplied through standard input, and waits for your response to the prompt.

For information about NFS V4 ACLs, see the topic *Managing GPFS access control lists* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Any output from the **mmgetac1** command can be used as input to **mmputac1**. The command is extended to support NFS V4 ACLs. In the case of NFS V4 ACLs, there is no concept of a default ACL. Instead, there is a single ACL and the individual access control entries can be flagged as being inherited (either by files, directories, both, or neither). Consequently, specifying the **-d** flag for an NFS V4 ACL is an error. By its nature, storing an NFS V4 ACL implies changing the inheritable entries (the GPFS default ACL) as well.

The following describes how **mmputac1** works for POSIX and NFS V4 ACLs:

Command	POSIX ACL	NFS V4 ACL
mmputac1	Access ACL (Error if default ACL is NFS V4 [1])	Stores the ACL (implies default as well)
mmputac1 -d	Default ACL (Error if access ACL is NFS V4 [1])	Error: NFS V4 ACL (has no default ACL)

[1] The default and access ACLs are not permitted to be mixed types because NFS V4 ACLs include inherited entries, which are the equivalent of a default ACL. An **mmde1ac1** of the NFS V4 ACL is required before an ACL is converted back to POSIX.

Depending on the file system's **-k** setting (**posix**, **nfs4**, or **all**), **mmputac1** may be restricted. The **mmputac1** command is not allowed to store an NFS V4 ACL if **-k posix** is in effect. The **mmputac1** command is not allowed to store a POSIX ACL if **-k nfs4** is in effect. For more information, see the description of the **-k** flag for the **mmchfs**, **mmcrfs**, and **mmlsfs** commands.

Note that the test to see if the given ACL is acceptable based on the file system's **-k** setting cannot be done until after the ACL is provided. For example, if **mmputac1 file1** is issued (no **-i** flag specified) the user then has to input the ACL before the command can verify that it is an appropriate ACL given the file system settings. Likewise, the command **mmputac1 -d dir1** (again the ACL was not given with the **-i** flag) requires that the ACL be entered before file system ACL settings can be tested. In this situation, the **-i** flag may be preferable to manually entering a long ACL, only to find out it is not allowed by the file system.

Parameters

Filename

The path name of the file or directory for which the ACL is to be set. If the **-d** option is specified, *Filename* must be the name of a directory.

Options

-d Specifies that the default ACL of a directory is to be set. This flag cannot be used on an NFS V4 ACL.

-i *InFilename*

The path name of a source file from which the ACL is to be read.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You may issue the **mmputacl** command only from a node in the GPFS cluster where the file system is mounted.

You must be the file or directory owner, the root user, or someone with control permission in the ACL, to run the **mmputacl** command.

Examples

To use the entries in a file named **standard.acl** to set the ACL for a file named **project2.history**, issue this command:

```
mmputacl -i standard.acl project2.history
```

where **standard.acl** contains:

```
user::rwx
group::rwx-
other:--x-
mask::rw-c
user:alpha:rwx
group:audit:rwx-
group:system:-w--
```

To confirm the change, issue this command:

```
mmgetacl project.history
```

The system displays information similar to:

```
#owner:paul
#group:design
user::rwx
group::rwx-
other:--x-
mask::rw-c
user:alpha:rwx
group:audit:rwx-
group:system:-w--
```

See also

- “mmeditACL command” on page 478
- “mmdelACL command” on page 446

mmpuac1

- “mmgetacl command” on page 500

Location

/usr/lpp/mmfs/bin

mmquotaoff command

Deactivates quota limit checking.

Synopsis

```
mmquotaoff [-u] [-g] [-j] [-v] {Device [Device ...] | -a}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmquotaoff** command disables quota limit checking by GPFS.

If none of: **-u**, **-j** or **-g** is specified, the **mmquotaoff** command deactivates quota limit checking for users, groups, and filesets.

If the **-a** option is not specified, *Device* must be the last parameter entered.

Parameters

Device [*Device* ...]

The device name of the file system to have quota limit checking deactivated.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Options

- a** Deactivates quota limit checking for all GPFS file systems in the cluster. When used in combination with the **-g** option, only group quota limit checking is deactivated. When used in combination with the **-u** or **-j** options, only user or fileset quota limit checking, respectively, is deactivated.
- g** Specifies that only group quota limit checking is to be deactivated.
- j** Specifies that only quota checking for filesets is to be deactivated.
- u** Specifies that only user quota limit checking is to be deactivated.
- v** Prints a message for each file system in which quotas are deactivated.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmquotaoff** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmquotaoff** command is issued.

mmquotaoff

Examples

1. To deactivate user quota limit checking on file system **fs0**, issue this command:

```
mmquotaoff -u fs0
```

To confirm the change, issue this command:

```
mmlsfs fs0 -Q
```

The system displays information similar to:

flag	value	description
-Q	group;fileset	Quotas enforced

2. To deactivate group quota limit checking on all file systems, issue this command:

```
mmquotaoff -g -a
```

To confirm the change, individually for each file system, issue this command:

```
mmlsfs fs2 -Q
```

The system displays information similar to:

flag	value	description
-Q	user;fileset	Quotas enforced

3. To deactivate all quota limit checking on file system **fs0**, issue this command:

```
mmquotaoff fs0
```

To confirm the change, issue this command:

```
mmlsfs fs0 -Q
```

The system displays information similar to:

flag	value	description
-Q	none	Quotas enforced

See also

- “mmcheckquota command” on page 361
- “mmdefquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmlsquota command” on page 559
- “mmquotaon command” on page 619
- “mmrepquota command” on page 627

Location

```
/usr/lpp/mmfs/bin
```


mmquotaon command

Activates quota limit checking.

Synopsis

```
mmquotaon [-u] [-g] [-j] [-v] {Device [Device...]} | -a}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmquotaon** command enables quota limit checking by GPFS.

If none of: **-u**, **-j** or **-g** is specified, the **mmquotaon** command activates quota limit checking for users, groups, and filesets.

If the **-a** option is not used, *Device* must be the last parameter specified.

After quota limit checking has been activated by issuing the **mmquotaon** command, issue the **mmcheckquota** command to count inode and space usage.

Parameters

Device [*Device...*]

The device name of the file system to have quota limit checking activated.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Options

- a** Activates quota limit checking for all of the GPFS file systems in the cluster. When used in combination with the **-g** option, only group quota limit checking is activated. When used in combination with the **-u** or **-j** option, only user or fileset quota limit checking, respectively, is activated.
- g** Specifies that only group quota limit checking is to be activated.
- j** Specifies that only fileset quota checking is to be activated.
- u** Specifies that only user quota limit checking is to be activated.
- v** Prints a message for each file system in which quota limit checking is activated.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmquotaon** command.

mmquotaon

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmquotaon** command is issued.

Examples

1. To activate user quotas on file system **fs0**, issue this command:

```
mmquotaon -u fs0
```

To confirm the change, issue this command:

```
mmlsfs fs0 -Q
```

The system displays information similar to:

flag value	description
-Q user	Quotas enforced

2. To activate group quota limit checking on all file systems, issue this command:

```
mmquotaon -g -a
```

To confirm the change, individually for each file system, issue this command:

```
mmlsfs fs1 -Q
```

The system displays information similar to:

flag value	description
-Q group	Quotas enforced

3. To activate user, group, and fileset quota limit checking on file system **fs2**, issue this command:

```
mmquotaon fs2
```

To confirm the change, issue this command:

```
mmlsfs fs2 -Q
```

The system displays information similar to:

flag value	description
-Q user;group;fileset	Quotas enforced

See also

- “mmcheckquota command” on page 361
- “mmdefquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmlsquota command” on page 559
- “mmquotaoff command” on page 617
- “mmrepquota command” on page 627

Location

```
/usr/lpp/mmfs/bin
```

mmremoteclassifier command

Manages information about remote GPFS clusters.

Synopsis

```
mmremoteclassifier add RemoteClusterName [-n ContactNodes] [-k KeyFile]
```

or

```
mmremoteclassifier update RemoteClusterName [-C NewClusterName] [-n ContactNodes] [-k KeyFile]
```

or

```
mmremoteclassifier delete {RemoteClusterName | all}
```

or

```
mmremoteclassifier show [RemoteClusterName | all]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmremoteclassifier** command is used to make remote GPFS clusters known to the local cluster, and to maintain the attributes associated with those remote clusters. The keyword appearing after **mmremoteclassifier** determines which action is performed:

add

Adds a remote GPFS cluster to the set of remote clusters known to the local cluster.

delete

Deletes the information for a remote GPFS cluster.

show

Displays information about a remote GPFS cluster.

update

Updates the attributes of a remote GPFS cluster.

To be able to mount file systems that belong to some other GPFS cluster, you must first make the nodes in this cluster aware of the GPFS cluster that owns those file systems. This is accomplished with the **mmremoteclassifier add** command. The information that the command requires must be provided to you by the administrator of the remote GPFS cluster. You will need this information:

- The name of the remote cluster.
- The names or IP addresses of a few nodes that belong to the remote GPFS cluster.
- The public key file generated by the administrator of the remote cluster by running the **mmauth genkey** command for the remote cluster.

Since each cluster is managed independently, there is no automatic coordination and propagation of changes between clusters like there is between the nodes within a cluster. This means that once a remote cluster is defined with the **mmremoteclassifier** command, the information about that cluster is automatically propagated across all nodes that belong to this cluster. But if the administrator of the remote cluster decides to rename it, or deletes some or all of the contact nodes, or change the public key file, the information in this cluster becomes obsolete. It is the responsibility of the administrator of the remote GPFS cluster to notify you of such changes so that you can update your information using the appropriate options of the **mmremoteclassifier update** command.

mmremoteclass

Parameters

RemoteClusterName

Specifies the cluster name associated with the remote cluster that owns the remote GPFS file system. The value **all** indicates all remote clusters defined to this cluster, when using the **mmremoteclass delete** or **mmremoteclass show** commands.

-C *NewClusterName*

Specifies the new cluster name to be associated with the remote cluster.

-k *KeyFile*

Specifies the name of the public key file provided to you by the administrator of the remote GPFS cluster.

-n *ContactNodes*

A comma separated list of nodes that belong to the remote GPFS cluster, in this format:

```
[tcpPort=NNNN,]node1[,node2 ...]
```

where:

```
tcpPort=NNNN
```

Specifies the TCP port number to be used by the local GPFS daemon when contacting the remote cluster. If not specified, GPFS will use the default TCP port number 1191.

```
node1[,node2...]
```

Specifies a list of nodes that belong to the remote cluster. The nodes can be identified through their host names or IP addresses.

Exit status

0 Successful completion. After successful completion of the **mmremoteclass** command, the new configuration information is propagated to all nodes in the cluster.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmremoteclass** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. This command adds remote cluster **k164.kgn.ibm.com** to the set of remote clusters known to the local cluster, specifying **k164n02** and **k164n03** as remote contact nodes. File **k164.id_rsa.pub** is the name of the public key file provided to you by the administrator of the remote cluster.

```
mmremoteclass add k164.kgn.ibm.com -n k164n02,k164n03 -k k164.id_rsa.pub
```

The output is similar to this:

```
mmremoteclass: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

2. This command displays information for the remote cluster **k164.kgn.ibm.com**.

```
mmremoteclass show k164.kgn.ibm.com
```

The output is similar to this:

```
Cluster name:    k164.kgn.ibm.com
Contact nodes:  k164n02,k164n03
SHA digest:     a3917c8282fca7a27d951566940768dcd241902b
File systems:   (none defined)
```

For more information on the SHA digest, see the *IBM Spectrum Scale: Problem Determination Guide* and search on *SHA digest*.

- This command updates information for the remote cluster **k164.kgn.ibm.com**, changing the remote contact nodes to **k164n02** and **k164n01**. The TCP port to be used when contacting cluster **k164.kgn.ibm.com** is defined to be 6667.

```
mmremoteccluster update k164.kgn.ibm.com -n tcpPort=6667,k164n02,k164n01
```

The output is similar to this:

```
mmremoteccluster: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

The **mmremoteccluster show** command can then be used to see the changes.

```
mmremoteccluster show k164.kgn.ibm.com
```

The output is similar to this:

```
Cluster name:    k164.kgn.ibm.com
Contact nodes:   tcpPort=6667,k164n02,k164n01
SHA digest:      a3917c8282fca7a27d951566940768dcd241902b
File systems:    (none defined)
```

For more information on the SHA digest, see the *IBM Spectrum Scale: Problem Determination Guide* and search on *SHA digest*.

- This command deletes information for remote cluster **k164.kgn.ibm.com** from the local cluster.

```
mmremoteccluster delete k164.kgn.ibm.com
```

The output is similar to this:

```
mmremoteccluster: 6027-1371 Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
```

See also

- “mmauth command” on page 276
- “mmremotefs command” on page 624

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: Advanced Administration Guide*.

Location

```
/usr/lpp/mmfs/bin
```

mmremotefs command

Manages information needed for mounting remote GPFS file systems.

Synopsis

```
mmremotefs add Device -f RemoteDevice -C RemoteClusterName  
               [-T MountPoint] [-t DriveLetter]  
               [-A {yes | no | automount}] [-o MountOptions] [--mount-priority Priority]
```

or

```
mmremotefs delete {Device | all | -C RemoteClusterName} [--force]
```

or

```
mmremotefs show [Device | all | -C RemoteClusterName]
```

or

```
mmremotefs update Device [-f RemoteDevice] [-C RemoteClusterName]  
                    [-T MountPoint] [-t DriveLetter]  
                    [-A {yes | no | automount}] [-o MountOptions] [--mount-priority Priority]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmremotefs** command is used to make GPFS file systems that belong to other GPFS clusters known to the nodes in this cluster, and to maintain the attributes associated with these file systems. The keyword appearing after **mmremotefs** determines which action is performed:

add

Define a new remote GPFS file system.

delete

Delete the information for a remote GPFS file system.

show

Display the information associated with a remote GPFS file system.

update

Update the information associated with a remote GPFS file system.

Use the **mmremotefs** command to make the nodes in this cluster aware of file systems that belong to other GPFS clusters. The cluster that owns the given file system must have already been defined with the **mmremotefcluster** command. The **mmremotefs** command is used to assign a local name under which the remote file system will be known in this cluster, the mount point where the file system is to be mounted in this cluster, and any local mount options that you may want.

Once a remote file system has been successfully defined and a local device name associated with it, you can issue normal commands using that local name, the same way you would issue them for file systems that are owned by this cluster.

When running the **mmremotefs** command delete and update options, the file system must be unmounted on the local cluster. However, it can be mounted elsewhere.

Parameters

Device

Specifies the name by which the remote GPFS file system will be known in the cluster.

-C *RemoteClusterName*

Specifies the name of the GPFS cluster that owns the remote GPFS file system.

-f *RemoteDevice*

Specifies the actual name of the remote GPFS file system. This is the device name of the file system as known to the remote cluster that owns the file system.

Options

-A {**yes** | **no** | **automount**}

Indicates when the file system is to be mounted:

yes

When the GPFS daemon starts.

no Manual mount. This is the default.

automount

When the file system is first accessed.

-o *MountOptions*

Specifies the mount options to pass to the mount command when mounting the file system. For a detailed description of the available mount options, see “GPFS-specific mount options” on page 28 in *IBM Spectrum Scale: Administration and Programming Reference*.

-T *MountPoint*

The local mount point directory of the remote GPFS file system. If it is not specified, the mount point will be set to *DefaultMountDir/Device*. The default value for *DefaultMountDir* is */gpfs*, but it can be changed with the **mmchconfig** command.

-t *DriveLetter*

Specifies the drive letter to use when the file system is mounted on Windows.

--mount-priority *Priority*

Controls the order in which the individual file systems are mounted at daemon startup or when one of the **all** keywords is specified on the **mmmout** command.

File systems with higher *Priority* numbers are mounted after file systems with lower numbers. File systems that do not have mount priorities are mounted last. A value of zero indicates no priority.

--force

The **--force** flag can only be used with the delete option. It will override an error that can occur when trying to delete a remote mount where the remote cluster was already removed. If the original delete attempt returns an error stating it cannot check to see if the mount is in use, then this is the condition to use. The **--force** flag overrides and allows the deletion to complete.

Exit status

0 Successful completion. After successful completion of the **mmremotefs** command, the new configuration information is propagated to all nodes in the cluster.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmremotefs** command.

mmremotefs

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

This command adds remote file system **gpfsn**, owned by remote cluster **k164.kgn.ibm.com**, to the local cluster, assigning **rgpfsn** as the local name for the file system, and **/gpfs/rgpfsn** as the local mount point.

```
mmremotefs add rgpfsn -f gpfsn -C k164.kgn.ibm.com -T /gpfs/rgpfsn
```

The output is similar to this:

```
mmremotefs: 6027-1371 Propagating the cluster configuration data to all
  affected nodes. This is an asynchronous process.
```

The **mmremotefs show** command can be used to see the changes.

```
mmremotefs show rgpfsn
```

The output is similar to this:

Local Name	Remote Name	Cluster name	Mount Point	Mount Options	Automount	Drive
rgpfs1	gpfs1	gpfs-n60-win.fvtomain.net	/rgpfs1	rw	no	K

See also

- “mmauth command” on page 276
- “mmremotecluster command” on page 621

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: Advanced Administration Guide*.

Location

```
/usr/lpp/mmfs/bin
```


mmrepquota command

Displays file system user, group, and fileset quotas.

Synopsis

```
mmrepquota [-u] [-g] [-e] [-q] [-n] [-v] [-t]
            [--block-size {BlockSize | auto}] {-a | Device:Fileset ...}
```

or

```
mmrepquota [-u] [-g] [-j] [-e] [-q] [-n] [-v] [-t]
            [--block-size {BlockSize | auto}] {-a | Device...}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmrepquota** command reports file system usage and quota information for a user, group, or fileset.

This command cannot be run from a Windows node.

If **-g**, **-j**, or **-u** are not specified, then user, group, and fileset quotas are listed.

If **-a** is not specified, *Device* must be the last parameter entered.

For each file system in the cluster, the **mmrepquota** command displays:

1. Block limits (displayed in number of data blocks in 1KB units or a unit defined by the **--block-size** parameter):
 - quota type (USR, GRP or FILESET)
 - current usage (the amount of disk space used by this user, group, or fileset, in 1KB units or a unit defined by the **--block-size** parameter)
 - soft limit (the amount of disk space that this user, group, or fileset is allowed to use during normal operation, in 1KB units or a unit defined by the **--block-size** parameter)
 - hard limit (the total amount of disk space that this user, group, or fileset is allowed to use during the grace period, in 1KB units or a unit defined by the **--block-size** parameter)
 - space in doubt
 - grace period
2. File limits:
 - current number of files
 - soft limit
 - hard limit
 - files in doubt
 - grace period

Note: In cases where small files do not have an additional block allocated for them, quota usage may show less space usage than expected.

3. Entry Type

default on

Default quotas are enabled for this file system.

mmrepquota

default off

Default quotas are not enabled for this file system.

e Explicit quota limits have been set using the **mmedquota** command.

d_fsys The quota limits are the default file system values set using the **mmdefedquota** command.

d_fset The quota limits are the default fileset-level values set using the **mmdefedquota** command.

i Default quotas were not enabled when this initial entry was established. Initial quota limits have a value of zero indicating no limit.

Because the sum of the in-doubt value and the current usage may not exceed the hard limit, the actual block space and number of files available to the user, group, or fileset may be constrained by the *in-doubt* value. If the *in-doubt* values approach a significant percentage of the quota, run the **mmcheckquota** command to account for the lost space and files.

See “Listing quotas” on page 177 in *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

Device

The device name of the file system to be listed.

If more than one file system is listed, the names must be delimited by a space. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

Fileset

Specifies an optional fileset to be listed.

- a** Lists quotas for all file systems in the cluster. A header line is printed automatically with this option.
- e** Specifies that the **mmrepquota** command is to collect updated quota usage data from all nodes before displaying results. If this option is not specified, there is the potential to display negative usage values as the quota server may process a combination of up-to-date and back-level information.
- g** Lists only group quotas.
- j** Lists only fileset quotas.
- n** Displays a numerical user ID.
- q** Shows whether quota enforcement is active.
- t** Lists global user, group, and fileset block and inode grace times.
- u** Lists only user quotas.
- v** Prints a header line.
- block-size {BlockSize | auto}**
Specifies the unit in which the number of blocks is displayed. The value must be of the form **[n]K**, **[n]M**, **[n]G** or **[n]T**, where *n* is an optional integer in the range 1 to 1023. The default is 1K. If **auto** is specified, the number of blocks is automatically scaled to an easy-to-read value.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrepquota** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

GPFS must be running on the node from which the **mmrepquota** command is issued.

Examples

1. To report on user quotas for file system **fs2** and display a header line, issue this command:

```
mmrepquota -u -v fs2
```

The system displays information similar to:

```
*** Report for USR quotas on fs2
      Block Limits          |          File Limits
      in                    |          in          entry
Name  type  KB  quota  limit  doubt  grace | files  quota  limit  doubt  grace  Type
root  USR   8   0     0     0     none | 1     0     0     0     none  default  on
user2 USR 2016 256 512  0     6days | 7    10    20    0     none  d_fsys
user3 USR 104 256 512  0     none   | 1    10    20    0     none  d_fsys
user4 USR  0 256 512  0     none   | 0    10    20    0     none  d_fsys
user5 USR 368 256 512  0 23hours | 5     4    10    0 23hours d_fsys
user6 USR  0 256 512  0     none   | 0    10    20    0     none  d_fsys
user7 USR 1024 1024 5120 4096 none | 1     0     0    19    none  e
```

2. To report on quota enforcement for **fs2**, issue this command:

```
mmrepquota -q fs2
```

The system displays information similar to:

```
fs2: USR quota is on; default quota is on
fs2: GRP quota is on; default quota is on
fs2: FILESET quota is on; default quota is off
```

3. To report on user quotas for file system **gpfs2**, issue this command:

```
mmrepquota -u gpfs2
```

The system displays information similar to:

```
      Block Limits          |          File Limits
      in                    |          in
Name  fileset type  KB  quota  limit  doubt  grace | files  quota  limit  doubt  grace
root  root   USR   0   0     0     0     none | 1     0     0     0     none
root  fset3  USR   8   0     0     0     none | 1     0     0     0     none
root  fset4  USR 8192  0     0     0     none | 1     0     0     0     none
pfs001 root   USR   0 10240  0     0     none | 0    1000  5000  0     none
pfs001 fset3  USR   0 10240  0     0     none | 0    1000  5000  0     none
pfs001 fset4  USR 4104 10240 153600 0     none | 2    1000  5000  0     none
```

4. To report on user quotas for file system **gpfs2** in fileset **fset4**, issue this command:

```
mmrepquota -u gpfs2:fset4
```

The system displays information similar to:

```
      Block Limits          |          File Limits
      in                    |          in
Name  fileset type  KB  quota  limit  doubt  grace | files  quota  limit  doubt  grace
root  fset4  USR 8192  0     0     0     none | 1     0     0     0     none
pfs001 fset4  USR 4104 10240 153600 0     none | 2    1000  5000  0     none
```

5. To list global user, group, and fileset block and inode grace times, issue this command:

```
mmrepquota -u -t gpfs_s
```

The system displays information similar to:

mmrepquota

```
User:   block default grace time 7days, inode default grace time 7days
Group:  block default grace time 7days, inode default grace time 7days
Fileset: block default grace time 7days, inode default grace time 7days
```

Block Limits								File Limits				
Name	type	KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace	
root	USR	0	0	0	0	none	50	0	0	0	none	
ftp	USR	0	0	0	0	none	50	0	0	0	none	

6. To report on fileset quotas in file system fs1, issue this command:

```
mmrepquota -j fs1 --block-size auto
```

The system displays information similar to:

Block Limits								File Limits				
Name	fileset	type	blocks	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace
root	root	FILESET	256K	0	0	0	none	1	0	0	0	none
fset0	root	FILESET	89.25G	100G	200G	89.99M	none	13729	4000	5000	0	7 days
fset1	root	FILESET	0	100G	200G	0	none	1	4000	5000	0	none

Note: In any **mmrepquota** listing, when the 'type' is FILESET, the 'Name' column heading is meant to indicate the fileset name (root, fset0, fset1, in this example), and the value for the 'fileset' column heading (root, in this example) can be ignored.

See also

- “mmcheckquota command” on page 361
- “mmdefquota command” on page 434
- “mmdefquotaoff command” on page 437
- “mmdefquotaon command” on page 440
- “mmedquota command” on page 481
- “mmlsquota command” on page 559
- “mmquotaoff command” on page 617
- “mmquotaon command” on page 619

Location

```
/usr/lpp/mmfs/bin
```

mmrestoreconfig command

Restores file system configuration information.

Synopsis

```
mmrestoreconfig Device -i InputFile [-I {yes | test}]
[-Q {yes | no | only}] [-W NewDeviceName]
```

or

```
mmrestoreconfig Device -i InputFile --image-restore
[-I {yes | test}] [-W NewDeviceName]
```

or

```
mmrestoreconfig Device -i InputFile -F QueryResultFile
```

or

```
mmrestoreconfig Device -i InputFile -I continue
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

The **mmrestoreconfig** command allows you to either query or restore, or both query *and* restore, the output file of the **mmbackupconfig** command.

In the **query phase**, the **mmrestoreconfig** command uses the output file generated by the **mmbackupconfig** command as an input parameter, and then creates a configuration file. Users can then edit the configuration file to fit their current file system configuration. You can use the definitions in the configuration file to create the appropriate network shared disks (NSDs) and file systems required for the restore.

In the **image restore phase**, the **mmrestoreconfig** command uses the input file (output from the **mmbackupconfig** command) to restore the backed up file system configuration in the newly created file system. The newly created file system must not be mounted prior to the **mmimgrestore** command execution thus the quota settings are turned off for image restore. They can be reactivated after the **mmimgrestore** command is completed using the **-Q only** flag of the **mmrestoreconfig** command.

This command cannot be run from a Windows node.

Parameters

Device

Specifies the name of the file system to be restored.

-i *inputFile*

Specifies the file generated by the **mmbackupconfig** command. The input file contains the file system configuration information.

-I {yes | **test**}

Specifies the action to be taken during the restore phase:

yes

Test and proceed on the restore process. This is the default action.

test

Test all the configuration settings before the actual restore is performed.

mmrestoreconfig

Use **-I continue** to restart **mmrestoreconfig** from the last known successful configuration restore.

-F *QueryResultFile*

Specifies the pathname of the configuration query result file generated by **mmrestoreconfig**. The configuration query result file is a report file that you can edit and use as a guide to **mmcrnsd** or **mmcrfs**.

--image-restore

Restores the configuration data in the proper format for Scale Out Backup and Restore (SOBAR).

-Q {yes | no | only}

Specifies whether quota settings are enforced during the file system restore. If set to **no**, the quota settings are ignored.

To restore quota settings after the **mmimgrestore** command has successfully run, the **-Q only** option must be specified.

-W *newDeviceName*

Restores the backed up file system information to this new device name

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrestoreconfig** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. Run **mmrestoreconfig -F QueryResultFile** to specify the pathname of the configuration query result file to be generated.

```
mmrestoreconfig gpfs1 -i inputFile -F reportfile
```

2. To test settings before running a restore:

```
mmrestoreconfig fs1 -i /tmp/fs1.mmbackupconfig.out -I test
```

The system displays output similar to:

```
-----  
Configuration test restore of fs1 begins at Wed Mar 14 16:00:16 EDT 2012.  
-----
```

```
mmrestoreconfig: Checking disk settings for fs1:  
mmrestoreconfig: Checking the number of storage pools defined for fs1.  
The restored filesystem currently has 1 pools defined.  
mmrestoreconfig: Checking storage pool names defined for fs1.  
Storage pool 'system' defined.  
mmrestoreconfig: Checking storage pool size for 'system'.  
mmrestoreconfig: Storage pool size 127306752 was defined for 'system'.
```

```
mmrestoreconfig: Checking filesystem attribute configuration for fs1:  
File system attribute to be restored: defaultDataReplicas  
Backup value: 2  
Current value: 1
```

```
mmrestoreconfig: Checking fileset configurations for fs1:
```

```
Fileset to restore: root.
Fileset status: Linked /fs1
Fileset mode: off
Fileset to restore: smallfileset.
Fileset status: Linked /fs1/smallfileset
Fileset mode: off
```

```
mmrestoreconfig: Checking policy rule configuration for fs1:
mmrestoreconfig: Testing policy configuration restore.
Validated policy `policyfile.backup': parsed 1 Placement Rules, 0 Restore Rules,
    0 Migrate/Delete/Exclude Rules,
    0 List Rules, 0 External Pool/List Rules
```

```
mmrestoreconfig: Checking quota settings for fs1:
mmrestoreconfig: Checking quota enablement for fs1.
```

```
mmrestoreconfig: Disabling the following settings:
mmrestoreconfig: Enabling the following settings: -u -g -j
```

```
mmrestoreconfig: Disabling the following default quota settings: -u -g -j
```

```
mmrestoreconfig: Enabling the following default quota settings: -u -g -j
```

```
mmrestoreconfig: Quota limits for fs1:
```

```
mmrestoreconfig: Default Quota limits for fs1:
```

```
mmrestoreconfig: Command successfully completed
```

3. Run **mmrestoreconfig** to restore the **gpfs1** file system:

```
mmrestoreconfig gpfs1 -i inputFile
```

4. To restore the **fs9** file system configuration data prior to the **mmimgrestore** command, issue:

```
mmrestoreconfig fs9 -i fs9.backupconfig --image-restore
```

The system displays output similar to:

```
mmrestoreconfig: Quota and DMAPI are enabled.
mmrestoreconfig: Disabling quota and/or DMAPI ...
-----
Configuration restore of fs9 begins at Thu Nov 29 17:09:55 EST 2012.
```

```
-----
mmrestoreconfig: Checking disk settings for fs9:
mmrestoreconfig: Checking the number of storage pools defined for fs9.
mmrestoreconfig: Checking storage pool names defined for fs9.
mmrestoreconfig: Checking storage pool size for 'system'.
```

```
mmrestoreconfig: Checking filesystem attribute configuration for fs9:
```

```
mmrestoreconfig: Checking policy rule configuration for fs9:
mmrestoreconfig: No policy rules installed in backed up filesystem fs9.
mmrestoreconfig: Command successfully completed
```

5. To restore the quota settings for file system **fs9**, after the **mmimgrestore** command, issue:

```
mmrestoreconfig fs9 -i fs9.backupconfig -Q only
```

The system displays output similar to:

```
-----
Configuration restore of fs9 begins at Thu Nov 29 17:13:51 EST 2012.
```

```
-----
mmrestoreconfig: Checking quota settings for fs9:
mmrestoreconfig: Checking quota enablement for fs9.
```

```
mmrestoreconfig: Restoring quota and defquota limits for fs9:
fs9: Start quota check
    11 % complete on Thu Nov 29 17:17:37 2012
```

mmrestoreconfig

```
22 % complete on Thu Nov 29 17:17:37 2012
33 % complete on Thu Nov 29 17:17:37 2012
44 % complete on Thu Nov 29 17:17:37 2012
55 % complete on Thu Nov 29 17:17:38 2012
69 % complete on Thu Nov 29 17:17:38 2012
84 % complete on Thu Nov 29 17:17:38 2012
100 % complete on Thu Nov 29 17:17:39 2012
Finished scanning the inodes for fs9.
Merging results from scan.
mmrestoreconfig: Command successfully completed
```

See also

- “mmbackupconfig command” on page 290
- “mmimgbackup command” on page 508
- “mmimgrestore command” on page 511

Location

/usr/lpp/mmfs/bin

mmrestorefs command

Restores a file system or an independent fileset from a snapshot.

Synopsis

```
mmrestorefs Device SnapshotName [-j FilesetName]
          [-N {Node[,Node...] | NodeFile | NodeClass}]
          [--log-quiet] [--preserve-encryption-attributes]
          [--suppress-external-attributes] [--threads MaxNumThreads]
          [--work-unit FilesPerThread]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher. Available on AIX and Linux.

Description

Use the **mmrestorefs** command to restore user data and attribute files to a file system or an independent fileset using those of the specified snapshot. Data will be restored by **mmrestorefs** without regard for file system or fileset quotas unless the **enforceFilesetQuotaOnRoot** configuration attribute of the **mmchconfig** command is set to **yes**. The **mmrestorefs** command does not restore the file system and fileset quota configuration information.

In IBM Spectrum Scale 4.1.1 and earlier, ensure that the file system is mounted before you run the **mmrestorefs** command. For more information, see Table 27 on page 636 and Table 28 on page 636 below. When restoring from an independent fileset snapshot (using the **-j** option), link the fileset from nodes in the cluster that are to participate in the restore. It is preferable to run the **mmrestorefs** command when there are no user operations (either from commands, applications, or services) in progress on the file system or fileset. If there are user operations in progress on the file system or fileset while **mmrestorefs** is running, the restore might fail. For these failures, stop the user operations and run the **mmrestorefs** command again to complete the restore. For better performance, run the **mmrestorefs** command when the system is idle. While the restore is in progress, do not unlink the fileset, unmount the file system, or delete the fileset, fileset snapshot, or file system.

The **mmrestorefs** command cannot restore a fileset that was deleted after a global snapshot was created. In addition, the filesets in a global snapshot that are in deleted or unlinked state cannot be restored.

Snapshots are not affected by the **mmrestorefs** command. When a failure occurs during a restore, try repeating the **mmrestorefs** command except when there are **ENOSPC** or quota exceeded errors. In these cases, fix the errors then try the **mmrestorefs** command again.

For information on how GPFS policies and snapshots interact, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Because snapshots are not copies of the entire file system, they should not be used as protection against media failures. For protection against media failures, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and search on “recoverability considerations”.

The **mmrestorefs** command can cause a compressed file in the active file system to become decompressed if it is overwritten by the restore process. To recompress the file, run the **mmrestripefile** command with the **-z** option.

mmrestorefs

CAUTION:

- Do not run file compression or decompression while an mmrestorefs command is running. This caution applies to compression or decompression with the mmchattr command or with the mmapplypolicy command.
- Do not run the mmrestripefs or mmrestripefile command while an mmrestorefs command is running.

Note: The following table shows the requirements and the results when you restore a global snapshot:

Table 27. Restoring a global snapshot

The product version level of the node that runs the mmrestorefs command	The file system must be in this state	Results	
Before V4.1.1	Unmounted	The file system manager performs the restore.	
V4.1.1 or later	Mounted	By default the restore is performed on all nodes running V4.1.1 or later.	

The following table shows the requirements and the results when you restore a fileset snapshot:

Table 28. Restoring a fileset snapshot

The product version level of the node that runs the mmrestorefs command	The file system must be in this state	Results	
V3.5	Unmounted	The file system manager performs the restore.	
V4.1.1 or later	Mounted	By default the restore is performed on all nodes running V4.1.1 or later.	

Parameters

Device

The device name of the file system that contains the snapshot to use for the restore. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

This must be the first parameter.

SnapshotName

Specifies the name of the snapshot that will be used for the restore.

-j FilesetName

Specifies the name of a fileset covered by this snapshot.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes that are to participate in the restore. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

Starting with IBM Spectrum Scale 4.1.1, **-N** can be used for both fileset and global snapshot restores. (In GPFS 4.1, **-N** can be used for fileset snapshot restore only. In GPFS 3.5 and earlier, there is no **-N** parameter.)

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--log-quiet

Suppresses detailed thread log output.

--preserve-encryption-attributes

Preserves the encryption extended attributes. Files that were removed after the snapshot was taken are restored with the same encryption attributes (including FEK) of the file in the snapshot. If this option is not used, the file is recreated with the encryption policy in place at the time the file is restored.

--suppress-external-attributes

Specifies that external attributes will not be restored.

--threads *MaxNumThreads*

Specifies the maximum number of concurrent restore operations. The default is 24.

--work-unit *FilesPerThread*

Specifies the number of files each thread will process at a time. The default is 100.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrestorefs** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

Suppose that you have the following directory structure:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3

/fs1/.snapshots/snap1/file1
/fs1/.snapshots/snap1/userA/file2
/fs1/.snapshots/snap1/userA/file3
```

The directory **userA** is then deleted, leaving the following structure:

```
/fs1/file1

/fs1/.snapshots/snap1/file1
/fs1/.snapshots/snap1/userA/file2
/fs1/.snapshots/snap1/userA/file3
```

The directory **userB** is then created using the inode originally assigned to **userA**, and another snapshot is taken:

```
mmcrsnapshot fs1 snap2
```

The output is similar to this:

```
Writing dirty data to disk.
Quiescing all file system operations.
Writing dirty data to disk again.
Snapshot snap2 created with id 2.
```

The directory structure is similar to the following:

mmrestorefs

```
/fs1/file1  
/fs1/userB/file2b  
/fs1/userB/file3b
```

```
/fs1/.snapshots/snap1/file1  
/fs1/.snapshots/snap1/userA/file2  
/fs1/.snapshots/snap1/userA/file3
```

```
/fs1/.snapshots/snap2/file1  
/fs1/.snapshots/snap2/userB/file2b  
/fs1/.snapshots/snap2/userB/file3b
```

The file system is then restored from **snap1**:

```
mmrestorefs fs1 snap1
```

The resulting directory structure is similar to the following:

```
/fs1/file1  
/fs1/userA/file2  
/fs1/userA/file3
```

```
/fs1/.snapshots/snap1/file1  
/fs1/.snapshots/snap1/userA/file2  
/fs1/.snapshots/snap1/userA/file3
```

```
/fs1/.snapshots/snap2/file1  
/fs1/.snapshots/snap2/userB/file2b  
/fs1/.snapshots/snap2/userB/file3b
```

See also

- “mmcrsnapshot command” on page 431
- “mmdelsnapshot command” on page 467
- “mmlssnapshot command” on page 563
- “mmsnapdir command” on page 674

Location

```
/usr/lpp/mmfs/bin
```

mmrestripefile command

Rebalances or restores the replication factor of the specified files, or performs any incomplete or deferred file compression or decompression.

Synopsis

```
mmrestripefile {-m | -r | -p | -b | -l | -z} {-F FilenameFile | Filename [Filename...]}
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmrestripefile** command attempts to repair the specified files, or performs any deferred or incomplete compression or decompression of the specified files. You can use **-F** option to specify a file that contains the list of file names to be processed, with one file name per line.

The repair options are rebalancing (**-b**), restoring replication factors (**-r**), migrating data (**-m**), and migrating file data to the proper pool (**-p**). The **-b** option not only rebalances files but also performs all the operations of the **-m** and **-r** options. For more information, see the topic "Restripping a GPFS file system" in the *IBM Spectrum Scale: Administration and Programming Reference*.

If you do not use replication, the **-r** and **-m** options are equivalent. Their behavior differs only on replicated files. After a successful rereplicate (**-r**) all suspended disks are empty. But a migrate operation (**-m**) leaves data on a suspended disk as long as at least one other replica of the data remains on a disk that is not suspended.

Use the **-l** option to relocate the block placement of the files.

Use the **-z** option to perform any deferred or incomplete compression or decompression of the files.

CAUTION:

Do not run the mmrestripefs or mmrestripefile command while an mmstorefs command is running.

Parameters

-F *FilenameFile*

Specifies a file that contains a list of names of files to be restriped, one name per line.

Filename

Specifies the names of one or more files to be restriped.

Options

-m Migrates critical data from any suspended disk for a list of specified files. Critical data is all data that would be lost if currently suspended disks were removed.

-r Migrates all data for a list of files from suspended disks. If a disk failure or removal makes some replicated data inaccessible, this command also restores replicated files to their designated level of replication. Use this option immediately after a disk failure to protect replicated data against a subsequent failure. You can also use this option before you take a disk offline for maintenance to protect replicated data against the failure of another disk during the maintenance process.

-p Directs **mmrestripefile** to repair the file placement within the storage pool.

Files that are assigned to one storage pool, but with data in a different pool, have their data migrated to the correct pool. These files are called ill-placed. Utilities, such as the **mmchattr** command, might change a file's storage pool assignment, but not move the data. The **mmrestripefile** command might

mmrestripefile

then be invoked to migrate all of the data at once, rather than migrating each file individually. The placement option (**-p**) rebalances only the files that it moves. In contrast, the rebalance operation (**-b**) performs data placement on all files.

- b** Rebalances a list of files across all disks that are not suspended, even if they are stopped. Although blocks are allocated on a stopped disk, they are not written to a stopped disk, nor are reads allowed from a stopped disk, until that disk is started and replicated data is copied onto it.
- l** Relocates the block placement of the file. The location of the blocks depends on the current write affinity depth, write affinity failure group setting, block group factor, and the node from which the command is run. For example, for an existing file, regardless of how its blocks are distributed on disks currently, if **mmrestripefile -l** is run from node A, the final block distribution looks as if the file was created from scratch on node A.
- z** Performs any deferred or incomplete compression or decompression of files. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrestripefile** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

This example illustrates restriping a file named `testfile0`. The following command confirms that the file is ill-placed:

```
mmfsattr -L testfile0
```

The system displays the following output:

```
file name:          testfile0
metadata replication: 2 max 2
data replication:   2 max 2
immutable:         no
appendOnly:        no
flags:             illplaced
storage pool name:  system
fileset name:      root
snapshot name:
```

To correct the problem, run the following command:

```
mmrestripefile -p testfile0
```

To confirm the change, run the following command:

```
mmfsattr -L testfile0
```

The system displays the following output:

```
file name:          testfile0
metadata replication: 2 max 2
data replication:   2 max 2
immutable:         no
appendOnly:        no
flags:
storage pool name:  system
fileset name:      root
snapshot name:
```

The following command compresses or decompresses a file for which compression or decompression is deferred or incomplete:

```
mmrestripefile -z largefile.data
```

See also

- “mmadddisk command” on page 239
- “mmapplypolicy command” on page 266
- “mmchattr command” on page 321
- “mmchdisk command” on page 354
- “mmdeldisk command” on page 449
- “mmrpldisk command” on page 648
- “mmrestripefs command” on page 642

Location

```
/usr/lpp/mmfs/bin
```

mmrestripefs command

Rebalances or restores the replication factor of all the files in a file system. Alternatively, this command performs any incomplete or deferred file compression or decompression of all the files in a file system.

Synopsis

```
mmrestripefs Device {-m | -r | -b | -p | -R | -c [--read-only] | -z} [-P PoolName]
                    [-N {Node[,Node...] | NodeFile | NodeClass}] [-o InodeResultFile]
                    [--inode-criteria CriteriaFile] [--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Run the **mmrestripefs** command to rebalance or restore the replication of all files in a file system. The command moves existing file system data between different disks in the file system based on changes to the disk state made by the **mmchdisk**, **mmadddisk**, and **mmdeldisk** commands. It also attempts to restore the metadata or data replication of all the files in the file system.

Alternatively, you can run the **mmrestripefs** command to perform any deferred or incomplete file compression or decompression in all the files of a file system.

You must specify one of the options (**-m**, **-r**, **-b**, **-R**, **-c**, **-p**, or **-z**) to indicate how much file system data to move or whether to perform file compression or decompression. You can run this command against a mounted or unmounted file system.

If the file system uses replication, then restriping the file system also replicates it. Also, if the file system uses replication the **-r** option and the **-m** options treat suspended disks differently. The **-r** option removes all data from a suspended disk. But the **-m** option leaves data on a suspended disk if at least one replica of the data remains on a disk that is not suspended.

The **-b** option performs all the operations of the **-m** and **-r** options.

Use the **-z** option to perform any deferred or incomplete file compression or decompression.

CAUTION:

Do not run the mmrestripefs or mmrestripefile command while an mmrestorefs command is running.

Consider the necessity of restriping and the current demands on the system. New data that is added to the file system is correctly striped. Restriping a large file system requires many insert and delete operations and might affect system performance. Plan to perform this task when system demand is low.

Parameters

Device

The device name of the file system to be restriped. File system names need not be fully qualified.

Device must be the first parameter. It can take the following parameters:

- m** Migrates all critical data off of any suspended disk in this file system. Critical data is all data that would be lost if currently suspended disks were removed.
- r** Migrates all data off suspended disks. It also restores all replicated files in the file system to their designated degree of replication when a previous disk failure or removal of a disk makes some replica data inaccessible. Use this parameter either immediately after a disk failure to protect

replicated data against a subsequent failure, or before you take a disk offline for maintenance to protect replicated data against failure of another disk during the maintenance process.

- b Rebalances all files across all disks that are not suspended, even if they are stopped. Although blocks are allocated on a stopped disk, they are not written to a stopped disk, nor are reads allowed from a stopped disk, until that disk is started and replicated data is copied onto it. The **mmrestripefs** command rebalances and restripes the file system. Use this option to rebalance the file system after you add, change, or delete disks in a file system.

Note: Rebalancing of files is an I/O intensive and time-consuming operation, and is important only for file systems with large files that are mostly invariant. In many cases, normal file update and creation will rebalance your file system over time, without the cost of the rebalancing.

- p Directs **mmrestripefs** to repair the file placement within the storage pool.

Files that are assigned to one storage pool, but with data in a different pool, have their data migrated to the correct pool. Such files are referred to as ill-placed. Utilities, such as the **mmchattr** command, might change a file's storage pool assignment, but not move the data. The **mmrestripefs** command might then be invoked to migrate all of the data at once, rather than migrating each file individually. The placement option (-p) rebalances only the files that it moves. In contrast, the rebalance operation (-b) performs data placement on all files.

- R Changes the replication settings of each file, directory, and system metadata object so that they match the default file system settings (see the **mmchfs** command -m and -r options) as long as the maximum (-M and -R) settings for the object allow it. Next, it replicates or unreplicates the object as needed to match the new settings. This option can be used to replicate all of the existing files that were not previously replicated or to unreplicate the files if replication is no longer needed or wanted.

- c Scans the file system and compares replicas of metadata and data for conflicts. When conflicts are found, the -c option attempts to fix the replicas.

--read-only

Modifies the -c option so that it does not try to fix conflicting replicas. You can use this option only with the -c option.

- z Performs any deferred or incomplete file compression or decompression of files in the file system. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-P PoolName

Directs **mmrestripefs** to repair only files assigned to the specified storage pool. This option is convenient for migrating ill-placed data blocks between pools, for example after you change a file's storage pool assignment with **mmchattr** or **mmapplypolicy** with the -I defer flag.

Do not use for other tasks, in particular, for any tasks that require metadata processing, such as re-replication. By design, all GPFS metadata is kept in the system pool, even for files that have blocks in other storage pools. Therefore a command that must process all metadata must not be restricted to a specific storage pool.

-N {Node[,Node...] | NodeFile | NodeClass}

Specify the nodes that participate in the restripe of the file system. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-o InodeResultFile

Contains a list of the inodes that met the interesting inode flags that were specified on the --inode-criteria parameter. The output file contains the following:

mmrestripfs

INODE_NUMBER

This is the inode number.

DISKADDR

Specifies a dummy address for later **tsfindinode** use.

SNAPSHOT_ID

This is the snapshot ID.

ISGLOBAL_SNAPSHOT

Indicates whether or not the inode is in a global snapshot. Files in the live file system are considered to be in a global snapshot.

INDEPENDENT_FSETID

Indicates the independent fileset to which the inode belongs.

MEMO (*INODE_FLAGS* *FILE_TYPE* [*ERROR*])

Indicates the inode flag and file type that will be printed:

Inode flags:

BROKEN
exposed
dataUpdateMiss
illCompressed
illPlaced
illReplicated
metaUpdateMiss
unbalanced

File types:

BLK_DEV
CHAR_DEV
DIRECTORY
FIFO
LINK
LOGFILE
REGULAR_FILE
RESERVED
SOCK
UNLINKED
DELETED

Notes:

1. An error message will be printed in the output file if an error is encountered when repairing the inode.
2. **DISKADDR**, **ISGLOBAL_SNAPSHOT**, and **FSET_ID** work with the **tsfindinode** tool (`/usr/lpp/mmfs/bin/tsfindinode`) to find the file name for each inode. **tsfindinode** uses the output file to retrieve the file name for each interesting inode.

--inode-criteria *CriteriaFile*

Specifies the interesting inode criteria flag, where *CriteriaFile* is one of the following:

BROKEN

Indicates that a file has a data block with all of its replicas on disks that have been removed.

Note: **BROKEN** is always included in the list of flags even if it is not specified.

dataUpdateMiss

Indicates that at least one data block was not updated successfully on all replicas.

exposed

Indicates an inode with an exposed risk; that is, the file has data where all replicas are on suspended disks. This could cause data to be lost if the suspended disks have failed or been removed.

i11Compressed

Indicates an inode in which file compression or decompression is deferred, or in which a compressed file is partly decompressed to allow the file to be written into or memory-mapped.

i11Placed

Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

i11Replicated

Indicates that the file has a data block that does not meet the setting for the replica.

metaUpdateMiss

Indicates that there is at least one metadata block that has not been successfully updated to all replicas.

unbalanced

Indicates that the file has a data block that is not well balanced across all the disks in all failure groups.

Note: If a file matches *any* of the specified interesting flags, all of its interesting flags (even those not specified) will be displayed.

--qos QoSClass

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrestripefs** command.

The node on which you run the command must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmrestripefs

Examples

1. To move all critical data from any suspended disk in file system **fs1**, run the following command:

```
mmrestripefs fs1 -m
```

The system displays information similar to the following output:

```
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...
8.00 % complete on Tue Feb 24 16:56:55 2009 ( 708608 inodes 346 MB)
100.00 % complete on Tue Feb 24 16:56:56 2009
GPFS: 6027-552 Scan completed successfully.
```

2. To rebalance all files in file system **fs1** across all defined, accessible disks that are not stopped or suspended, run the following command:

```
mmrestripefs fs1 -b
```

The system displays information similar to the following output:

```
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...
3.00 % complete on Tue Feb 24 16:56:39 2009 ( 180224 inodes 161 MB)
100.00 % complete on Tue Feb 24 16:56:44 2009
GPFS: 6027-552 Scan completed successfully.
```

3. To compare and fix replica conflicts of metadata and data in file system **gpfs1**, run the following command:

```
mmrestripefs gpfs1 -c
```

The system displays information similar to the following output:

```
Scanning file system metadata, phase 1 ...
Inode 0 in fileset 0 and snapshot 0 has mismatch in replicated disk address 2:104859136
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
100.00 % complete on Tue Jul 30 03:32:44 2013
Scan completed successfully.
```

4. To fix the pool placement of files in file system **fs1** and also determine which files are illReplicated (for example, as a result of a failed disk), run the following command:

```
mmrestripefs fs1 -p /tmp/crit --inode-criteria -o /tmp/inodeResultFile
```

The system displays information similar to the following output:

```
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
```

```

Scanning file system metadata for data storage pool
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...
  100.00 % complete on Wed Apr 15 10:15:15 2015 (65792 inodes with total 400 MB data processed)
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-3902 Check file '/tmp/inodeResultFile' on vmip1 for inodes that were \
                    found matching the criteria.
#10:15:15# vmip1:/fs1 # cat /tmp/crit
illReplicated
#10:15:19# vmip1:/fs1 # cat /tmp/inodeResultFile
This inode list was generated in the Parallel Inode Traverse on Wed Apr 15 10:15:14 2015
INODE_NUMBER DISKADDR SNAPSHOT_ID ISGLOBAL_SNAPSHOT FSET_ID MEMO(INODE_FLAGS FILE_TYPE [ERROR])
24320      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24322      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24321      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24324      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24325      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24323      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24326      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24327      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24328      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE
24329      0:0      0      1      0      illreplicated unbalanced REGULAR_FILE

```

See also

- “mmaddisk command” on page 239
- “mmapplypolicy command” on page 266
- “mmchattr command” on page 321
- “mmchdisk command” on page 354
- “mmchfs command” on page 371
- “mmdeldisk command” on page 449
- “mmrpldisk command” on page 648
- “mmrestripefile command” on page 639

Location

/usr/lpp/mmfs/bin

mmrpldisk command

Replaces the specified disk.

Synopsis

```
mmrpldisk Device DiskName {DiskDesc | -F StanzaFile} [-v {yes | no}]
[-N {Node[,Node...]} | NodeFile | NodeClass]
[--inode-criteria CriteriaFile] [-o InodeResultFile]
[--qos QOSClass]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmrpldisk** command to replace an existing disk in the GPFS file system with a new one. All data on the old disk is migrated to the new disk.

To replace a disk in a GPFS file system, you must first decide if you will:

1. Create a new disk using the **mmcrnsd** command.
In this case, use the rewritten disk stanza file produced by the **mmcrnsd** command or create a new disk stanza. When using the rewritten file, the disk usage and failure group specifications remain the same as specified on the **mmcrnsd** command.
2. Select a disk no longer in any file system. Issue the **mmlnsd -F** command to display the available disks.

The disk may then be used to replace a disk in the file system using the **mmrpldisk** command.

Notes:

1. You cannot replace a disk when it is the only remaining disk in the file system.
2. Under no circumstances should you replace a stopped disk. You need to start a stopped disk before replacing it. If a disk cannot be started, delete it using the **mmdeldisk** command. See the *IBM Spectrum Scale: Problem Determination Guide* and search for “Disk media failure”.
3. The file system may be mounted when running the **mmrpldisk** command.

Results

Upon successful completion of the **mmrpldisk** command, the disk is replaced in the file system and data is copied to the new disk without restriping.

Parameters

Device

The device name of the file system where the disk is to be replaced. File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

This must be the first parameter.

DiskName

The name of the disk to be replaced. To display the names of disks that belong to the file system, issue the **mmlnsd -f**, **mmlsfs -d**, or **mmlsdisk** command. The **mmlsdisk** command will also show the current disk usage and failure group values for each of the disks.

DiskDesc

A descriptor for the replacement disk.

Prior to GPFS 3.5, the disk information for the **mmrpldisk** command was specified in the form of a disk descriptor defined as follows (with the second, third, sixth, and seventh fields reserved):

```
DiskName:::DiskUsage:FailureGroup:::
```

For backward compatibility, the **mmrpldisk** command will still accept a traditional disk descriptor as input, but this use is discouraged.

-F StanzaFile

Specifies a file containing the NSD stanzas for the replacement disk. NSD stanzas have this format:

```
%nsd:
  nsd=NsdName
  usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}
  failureGroup=FailureGroup
  pool=StoragePool
  servers=ServerList
  device=DiskName
```

where:

nsd=NsdName

The name of an NSD previously created by the **mmcrnsd** command. For a list of available disks, issue the **mm lsnsd -F** command. This clause is mandatory for the **mmrpldisk** command.

usage={dataOnly | metadataOnly | dataAndMetadata | descOnly}

Specifies the type of data to be stored on the disk:

dataAndMetadata

Indicates that the disk contains both data and metadata. This is the default for disks in the system pool.

dataOnly

Indicates that the disk contains data and does not contain metadata. This is the default for disks in storage pools other than the system pool.

metadataOnly

Indicates that the disk contains metadata and does not contain data.

descOnly

Indicates that the disk contains no data and no file metadata. Such a disk is used solely to keep a copy of the file system descriptor, and can be used as a third failure group in certain disaster-recovery configurations. For more information, see the *IBM Spectrum Scale: Advanced Administration Guide* and search for “Synchronous mirroring utilizing GPFS replication”

This clause is optional for the **mmrpldisk** command. If omitted, the new disk will inherit the usage type of the disk being replaced.

failureGroup=FailureGroup

Identifies the failure group to which the disk belongs. A failure group identifier can be a simple integer or a topology vector that consists of up to three comma-separated integers. The default is -1, which indicates that the disk has no point of failure in common with any other disk.

GPFS uses this information during data and metadata placement to ensure that no two replicas of the same block can become unavailable due to a single failure. All disks that are attached to the same NSD server or adapter must be placed in the same failure group.

If the file system is configured with data replication, all storage pools must have two failure groups to maintain proper protection of the data. Similarly, if metadata replication is in effect, the system storage pool must have two failure groups.

Disks that belong to storage pools in which write affinity is enabled can use topology vectors to identify failure domains in a shared-nothing cluster. Disks that belong to traditional storage pools must use simple integers to specify the failure group.

mmrpldisk

This clause is optional for the **mmrpldisk** command. If omitted, the new disk will inherit the failure group of the disk being replaced.

pool=*StoragePool*

Specifies the storage pool to which the disk is to be assigned. This clause is ignored by the **mmrpldisk** command.

servers=*ServerList*

A comma-separated list of NSD server nodes. This clause is ignored by the **mmrpldisk** command.

device=*DiskName*

The block device name of the underlying disk device. This clause is ignored by the **mmrpldisk** command.

Note: While it is not absolutely necessary to specify the same parameters for the new disk as the old disk, it is suggested that you do so. If the new disk is equivalent in size to the old disk, and if the disk usage and failure group parameters are the same, the data and metadata can be completely migrated from the old disk to the new disk. A disk replacement in this manner allows the file system to maintain its current data and metadata balance.

If the new disk has a different size, disk usage, parameter, or failure group parameter, the operation may leave the file system unbalanced and require a restripe. Additionally, a change in size or the disk usage parameter may cause the operation to fail since other disks in the file system may not have sufficient space to absorb more data or metadata. In this case, first use the **mmaddisk** command to add the new disk, the **mmdelisk** command to delete the old disk, and finally the **mmrestripefs** command to rebalance the file system.

-v {**yes** | **no**}

Verify the new disk does not belong to an existing file system. The default is **-v yes**. Specify **-v no** only when you want to reuse a disk that is no longer needed for an existing file system. If the command is interrupted for any reason, use the **-v no** option on the next invocation of the command.

Important: Using **-v no** on a disk that already belongs to a file system will corrupt that file system. This will not be noticed until the next time that file system is mounted.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specify the nodes that participate in the migration of data from the old to the new disk. This command supports all defined node classes. The default is **all** or the current value of the **defaultHelperNodes** parameter of the **mmchconfig** command.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

--inode-criteria *CriteriaFile*

Specifies the interesting inode criteria flag, where *CriteriaFile* is one of the following:

BROKEN

Indicates that a file has a data block with all of its replicas on disks that have been removed.

Note: **BROKEN** is always included in the list of flags even if it is not specified.

dataUpdateMiss

Indicates that at least one data block was not updated successfully on all replicas.

exposed

Indicates an inode with an exposed risk; that is, the file has data where all replicas are on suspended disks. This could cause data to be lost if the suspended disks have failed or been removed.

i11Compressed

Indicates an inode in which file compression or decompression is deferred, or in which a compressed file is partly decompressed to allow the file to be written into or memory-mapped.

i11Placed

Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

i11Replicated

Indicates that the file has a data block that does not meet the setting for the replica.

metaUpdateMiss

Indicates that there is at least one metadata block that has not been successfully updated to all replicas.

unbalanced

Indicates that the file has a data block that is not well balanced across all the disks in all failure groups.

Note: If a file matches *any* of the specified interesting flags, all of its interesting flags (even those not specified) will be displayed.

-o InodeResultFile

Contains a list of the inodes that met the interesting inode flags that were specified on the **--inode-criteria** parameter. The output file contains the following:

INODE_NUMBER

This is the inode number.

DISKADDR

Specifies a dummy address for later **tsfindinode** use.

SNAPSHOT_ID

This is the snapshot ID.

ISGLOBAL_SNAPSHOT

Indicates whether or not the inode is in a global snapshot. Files in the live file system are considered to be in a global snapshot.

INDEPENDENT_FSETID

Indicates the independent fileset to which the inode belongs.

MEMO (INODE_FLAGS FILE_TYPE [ERROR])

Indicates the inode flag and file type that will be printed:

Inode flags:

BROKEN
exposed
dataUpdateMiss
i11Compressed
i11Placed
i11Replicated
metaUpdateMiss
unbalanced

File types:

BLK_DEV
CHAR_DEV
DIRECTORY
FIFO
LINK
LOGFILE
REGULAR_FILE

mmrpldisk

RESERVED
SOCK
UNLINKED
DELETED

Notes:

1. An error message will be printed in the output file if an error is encountered when repairing the inode.
2. **DISKADDR**, **ISGLOBAL_SNAPSHOT**, and **FSET_ID** work with the **tsfindinode** tool (`/usr/lpp/mmfs/bin/tsfindinode`) to find the file name for each inode. **tsfindinode** uses the output file to retrieve the file name for each interesting inode.

--qos *QoSClass*

Specifies the Quality of Service for I/O operations (QoS) class to which the instance of the command is assigned. If you do not specify this parameter, the instance of the command is assigned by default to the **maintenance** QoS class. This parameter has no effect unless the QoS service is enabled. For more information, see the topic *mmchqos command* in the *IBM Spectrum Scale: Administration and Programming Reference*. Specify one of the following QoS classes:

maintenance

This QoS class is typically configured to have a smaller share of file system IOPS. Use this class for I/O-intensive, potentially long-running GPFS commands, so that they contribute less to reducing overall file system performance.

other This QoS class is typically configured to have a larger share of file system IOPS. Use this class for administration commands that are not I/O-intensive.

For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmrpldisk** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To replace disk **hd27n01** in **fs1** with a new disk, **hd16vsdn10** allowing the disk usage and failure group parameters to default to the corresponding values of **hd27n01**, and have only nodes **c154n01**, **c154n02**, and **c154n09** participate in the migration of the data, issue this command:

```
mmrpldisk fs1 hd27n01 hd16vsdn10 -N c154n01,c154n02,c154n09
```

The system displays information similar to:

```
Replacing hd27n01 ...
```

```
The following disks of fs1 will be formatted on node c155n01.ppd.pok.ibm.com:
```

```
hd16vsdn10: size 17793024 KB
```

```
Extending Allocation Map
```

```
Checking Allocation Map for storage pool 'system'
```

```

 7 % complete on Wed May 16 16:36:30 2007
18 % complete on Wed May 16 16:36:35 2007
34 % complete on Wed May 16 16:36:40 2007
49 % complete on Wed May 16 16:36:45 2007
65 % complete on Wed May 16 16:36:50 2007
82 % complete on Wed May 16 16:36:55 2007
98 % complete on Wed May 16 16:37:00 2007
100 % complete on Wed May 16 16:37:01 2007
Completed adding disks to file system fs1.
Scanning system storage pool
Scanning file system metadata, phase 1 ...
 2 % complete on Wed May 16 16:37:04 2007
 7 % complete on Wed May 16 16:37:11 2007
14 % complete on Wed May 16 16:37:18 2007
20 % complete on Wed May 16 16:37:24 2007
27 % complete on Wed May 16 16:37:31 2007
34 % complete on Wed May 16 16:37:37 2007
50 % complete on Wed May 16 16:37:50 2007
61 % complete on Wed May 16 16:38:00 2007
68 % complete on Wed May 16 16:38:06 2007
79 % complete on Wed May 16 16:38:16 2007
90 % complete on Wed May 16 16:38:26 2007
100 % complete on Wed May 16 16:38:32 2007
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scanning file system metadata for fs1sp1 storage pool
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 3 % complete on Wed May 16 16:38:38 2007
25 % complete on Wed May 16 16:38:47 2007
53 % complete on Wed May 16 16:38:53 2007
87 % complete on Wed May 16 16:38:59 2007
97 % complete on Wed May 16 16:39:06 2007
100 % complete on Wed May 16 16:39:07 2007
Scan completed successfully.
Done
mmrpldisk: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.

```

- To replace disk **vmip3_nsd1** from storage pool **GOLD** on file system **fs2** and to search for any interesting files handled during the **mmrpldisk** at the same time, issue this command:

```
mmrpldisk fs2 vmip3_nsd1 -F f /tmp/crit --inode-criteria
```

The system displays information similar to:

```

Replacing vmip3_nsd1 ...

GPFS: 6027-531 The following disks of fs2 will be formatted on node vmip1:
    vmip2_nsd3: size 5120 MB
Extending Allocation Map
Checking Allocation Map for storage pool GOLD
 59 % complete on Wed Apr 15 10:52:44 2015
100 % complete on Wed Apr 15 10:52:49 2015
GPFS: 6027-1503 Completed adding disks to file system fs2.
GPFS: 6027-589 Scanning file system metadata, phase 1 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 2 ...
Scanning file system metadata for GOLD storage pool
Scanning file system metadata for BRONZE storage pool
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 3 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-589 Scanning file system metadata, phase 4 ...
GPFS: 6027-552 Scan completed successfully.
GPFS: 6027-565 Scanning user file metadata ...

```

mmrpldisk

```
6.47 % complete on Wed Apr 15 10:53:11 2015 ( 65792 inodes with total 448 MB data processed)
6.49 % complete on Wed Apr 15 10:55:01 2015 ( 65792 inodes with total 448 MB data processed)
100.00 % complete on Wed Apr 15 10:55:03 2015 ( 65792 inodes with total 448 MB data processed)
```

GPFS: 6027-552 Scan completed successfully.

GPFS: 6027-3902 Check file '/var/mmfs/tmp/fs2.pit.interestingInodes.12884901928' on vmip1 for inodes \ that were found matching the criteria.

Checking Allocation Map for storage pool GOLD

56 % complete on Wed Apr 15 10:55:08 2015

100 % complete on Wed Apr 15 10:55:12 2015

Done

mmrpldisk: 6027-1371 Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.

#11:57:08# vmip1:/fs2 # cat /tmp/crit

```
illReplicated
illPlaced
dataUpdateMiss
metaUpdateMiss
exposed
BROKEN
```

#11:09:24# vmip1:/fs2 # cat /var/mmfs/tmp/fs2.pit.interestingInodes.12884901928

This inode list was generated in the Parallel Inode Traverse on Wed Apr 15 10:55:02 2015

```
INODE_NUMBER DISKADDR SNAPSHOT_ID ISGLOBAL_SNAPSHOT FSET_ID MEMO(INODE_FLAGS FILE_TYPE [ERROR])
50177 0:0 0 1 0 illplaced REGULAR_FILE
```

Note: The **mmrpldisk** command will report any interesting inodes that it finds during routine processing, but the list might not be 100% accurate or complete.

See also

- “mmadddisk command” on page 239
- “mmchdisk command” on page 354
- “mmcrnsd command” on page 426
- “mmlsdisk command” on page 528
- “mmlsnsd command” on page 549
- “mmrestripefs command” on page 642

Location

/usr/lpp/mmfs/bin

mmsdrrestore command

Restores the latest GPFS system files on the specified nodes.

Synopsis

```
mmsdrrestore [-p nodeName] [-F mmsdrfsFile] [-R remoteFileCopyCommand]
             [-a | -N {Node[,Node...] | NodeFile | NodeClass}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmsdrrestore** command is intended for use by experienced system administrators.

Use the **mmsdrrestore** command to restore the latest GPFS system files on the specified nodes. If no nodes are specified, the command restores the configuration information only on the node on which it run. If the local GPFS configuration file is missing, the file that is specified with the **-F** option from the node that is specified with the **-p** option is used instead. This command works best when used with the **mmsdrbackup** user exit. See “mmsdrbackup user exit” on page 878 in *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

-p *nodeName*

Specifies the node from which to obtain a valid GPFS configuration file. The node must be either the primary configuration server or a node that has a valid backup copy of the **mmsdrfs** file. If not specified, the local node is used.

-F *mmsdrfsFile*

Specifies the path name of the GPFS configuration file for the **mmsdrrestore** command to use. This configuration file might be the current one on the primary server, or it might be a configuration file that is obtained from the **mmsdrbackup** user exit. If not specified, **/var/mmfs/gen/mmsdrfs** is used.

If the configuration file is a Cluster Configuration Repository (CCR) backup file, then you must also specify the **-a** option. All the nodes in the cluster are restored. However, if a configuration of the cluster is still available, you can restore the configuration of an individual node by running **mmsdrrestore -p**.

-R *remoteFileCopyCommand*

Specifies the fully qualified path name for the remote file copy program to be used for obtaining the GPFS configuration file. The default is **/usr/bin/rcp**.

-a Restores the GPFS configuration files on all nodes in the cluster.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Restores the GPFS configuration files on a set of nodes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*. This command does not support a *NodeClass* of mount.

Exit status

0 Successful completion.

nonzero

A failure occurred.

mmsdrrestore

Security

You must have root authority to run the **mmsdrrestore** command.

The node on which the command is issued must be able to run remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To restore the latest GPFS system files on the local node using the GPFS configuration file **/var/mmfs/gen/mmsdrfs** from the node that is named **primaryServer**, issue the following command:

```
mmsdrrestore -p primaryServer
```

The system displays output similar to the following example:

```
Tue Jul 3 18:19:53 CDT 2012: mmsdrrestore: Processing node k164n04.kgn.ibm.com
mmsdrrestore: Node k164n04.kgn.ibm.com successfully restored.
```

2. To restore the GPFS system files on all nodes in the cluster using GPFS configuration file **/GPFSconfigFiles/mmsdrfs.120605** on the node that is named **GPFSArchive**, issue the following command from the node named **localNode**:

```
mmsdrrestore -p GPFSArchive -F /GPFSconfigFiles/mmsdrfs.120605 -a
```

The system displays output similar to the following example:

```
Tue Jul 3 18:29:28 CDT 2012: mmsdrrestore: Processing node k164n04.kgn.ibm.com
Tue Jul 3 18:29:30 CDT 2012: mmsdrrestore: Processing node k164n05.kgn.ibm.com
Tue Jul 3 18:29:31 CDT 2012: mmsdrrestore: Processing node k164n06.kgn.ibm.com
mmsdrrestore: Command successfully completed
```

3. The following command restores the GPFS system files from the information in a Cluster Configuration Repository (CCR) backup file. When you restore from a CCR backup file, you must specify the **-a** option. All the nodes in the cluster are restored:

```
mmsdrrestore -F /GPFSbackupFiles/CCRBackup.2015.10.14.10.01.25.tar.gz -a
```

The command displays output similar to the following example:

```
Restoring CCR backup
CCR backup has been restored
```

See also

- “mmsdrbackup user exit” on page 878

Location

```
/usr/lpp/mmfs/bin
```

mmsetquota command

Sets quota limits.

Synopsis

```
mmsetquota Device{[:FilesetName]}
    [--user IdOrName[,IdOrName]] [--group IdOrName[,IdOrName]]}
    {[--block SoftLimit[:HardLimit]] [--files SoftLimit[:HardLimit]]}
```

or

```
mmsetquota Device[:FilesetName] --default {user | group}
    {[--block SoftLimit[:HardLimit]] [--files SoftLimit[:HardLimit]]}
```

or

```
mmsetquota Device --default fileset
    {[--block SoftLimit[:HardLimit]] [--files SoftLimit[:HardLimit]]}
```

or

```
mmsetquota Device --grace {user | group | fileset}
    {[--block GracePeriod] [--files GracePeriod]}
```

or

```
mmsetquota -F StanzaFile
```

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

The **mmsetquota** command sets quota limits, default quota limits, or grace periods for users, groups, and filesets in the specified file system.

When setting quota limits for a file system, replication within the file system should be considered. For explanation, see “Listing quotas” on page 177 in *IBM Spectrum Scale: Administration and Programming Reference*

Parameters

Device

Specifies the device name of the file system.

FilesetName

Specifies the name of a fileset located on *Device* for which quota information is to be set.

IdOrName

Specifies a numeric ID, user name, or group name.

SoftLimit

Specifies the amount of data or the number of files the user, group, or fileset will be allowed to use.

HardLimit

Specifies the amount of data or the number of files the user, group, or fileset will be allowed to use during a grace period. If omitted, the default is no limit. See note.

GracePeriod

Specifies the file-system grace period during which quotas can exceed the soft limit before it is imposed as a hard limit. See note.

mmsetquota

StanzaFile

Specifies a file containing quota stanzas.

--block

Specifies the quota limits or grace period for data.

--files

Specifies the quota limits or grace period for files.

--default

Sets the default quota for the user, group, or fileset.

--grace

Sets the grace period for the user, group, or fileset.

-F *StanzaFile*

Specifies a file containing the quota stanzas for set quota, set default quota, or set grace period.

Quota stanzas have this format:

```
%quota:  
  device=Device  
  command={setQuota|setDefaultQuota|setGracePeriod}  
  type={USR|GRP|FILESET}  
  id=IdList  
  fileset=FilesetName  
  blockQuota=Number  
  blockLimit=Number  
  blockGrace=Period  
  filesQuota=Number  
  filesLimit=Number  
  filesGrace=Period
```

where:

device=*Device*

The device name of file system.

command={setQuota|setDefaultQuota|setGracePeriod}

Specifies the command to be executed for this stanza.

setQuota

Sets the quota limits. This command ignores **blockGrace** and **filesGrace** attributes.

setDefaultQuota

Sets the default quota limits. This command ignores **id**, **blockGrace** and **filesGrace** attributes.

setGracePeriod

Sets the grace periods. The command ignores **id**, **fileset**, and quota limit attributes. Grace periods can be set for each quota type in the file system.

type={USR|GRP|FILESET}

Specifies whether the command applies to user, group, or fileset.

id=*IdList*

Specifies a list of numeric IDs or user, group, or fileset names.

fileset=*FilesetName*

Specifies the fileset name for the **perfileset** quota setting. This attribute is ignored for **type**=FILESET

blockQuota=*Number*

Specifies the block soft limit. The number can be specified using the suffix K, M, G, or T. See note.

blockLimit=Number

Specifies the block hard limit. The number can be specified using the suffix K, M, G, or T. See note.

filesQuota=Number

Specifies the inode soft limit. The number can be specified using the suffix K, M, or G. See note.

filesLimit=Number

Specifies the inode hard limit. The number can be specified using the suffix K, M, or G. See note.

blockGrace=Period

Specifies the file-system grace period during which the block quotas can exceed the soft limit before it is imposed as a hard limit. The period can be specified in days, hours, minutes, or seconds.

filesGrace=Period

Specifies the file-system grace period during which the files quota can exceed the soft limit before it is imposed as a hard limit. The period can be specified in days, hours, minutes, or seconds.

Note: The maximum block limit you can enter is 9999999999999999K. The maximum files limit you can enter is 2147483647.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmsetquota** command.

GPFS must be running on the node from which the **mmsetquota** command is issued.

You may issue the **mmsetquota** command only from a node in the GPFS cluster where the file system is mounted.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

- The following command sets the block soft and hard limit to 25G and 30G and files soft and hard limit to 10K and 11K, respectively for user user234:

```
# mmsetquota fs1 --user user234 --block 25G:30G --files 10K:11K
```

To verify the change, issue the following command:

```
# mmlsquota -u user234 fs1
```

Block Limits									File Limits				Remarks
Filesystem	Fileset	type	KB	quota	limit	in_doubt	grace	files	quota	limit	in_doubt	grace	
fs1	root	USR	143688	26214400	31457280	0	none	1	10240	11264	0	none	
fs1	mkfiles2	USR	no limits										
fs1	ifset1	USR	no limits										
fs1	1111	USR	no limits										
fs1	ifset2	USR	no limits										

- If perfilesset quota is enabled, the following command sets block soft and hard limit to 5G and 7G, respectively, for group fvt090 and for fileset ifset2:

```
# mmsetquota fs1:ifset2 --group fvt090 --block 5G:7G
```

To verify the change, issue the following command:

mmsetquota

```
# mmlsquota -g fvt090 fs1
```

```
Disk quotas for group fvt090 (gid 2590):
```

Filesystem	Fileset	Block Limits				File Limits					Remarks			
		type	KB	quota	limit	in_doubt	grace	files	quota	limit		in_doubt	grace	
fs1	root	GRP	none											
fs1	mkfiles2	GRP	none											
fs1	ifset1	GRP	none											
fs1	1111	GRP	none											
fs1	ifset2	GRP	143704	5242880	7340032	0	none	1	0	0	0	none		

3. To change the user grace period for block data to 10 days, issue the following command:

```
# mmsetquota fs1 --grace user --block 10days
```

4. All of the previous examples can be done in one invocation of **mmsetquota** by using quota stanza file. The stanza file `/tmp/quotaExample` may look like this:

```
%quota:  
device=fs1  
command=setquota  
type=USR  
id=user234  
blockQuota=25G  
blockLimit=30G  
filesQuota=10K  
filesLimit=11K
```

```
%quota:  
device=fs1  
command=setquota  
type=GRP  
id=fvt090  
fileset=ifset2  
blockQuota=5G  
blockLimit=7G
```

```
%quota:  
device=fs1  
command=setgraceperiod  
type=user  
blockGrace=10days
```

Then issue the command:

```
# mmsetquota -F /tmp/quotaExample
```

See also

- “`mmcheckquota` command” on page 361
- “`mmdefquota` command” on page 434
- “`mmdefquotaoff` command” on page 437
- “`mmdefquotaon` command” on page 440
- “`mmedquota` command” on page 481
- “`mmlsquota` command” on page 559
- “`mmquotaon` command” on page 619
- “`mmquotaoff` command” on page 617
- “`mmrepquota` command” on page 627

Location

```
/usr/lpp/mmfs/bin
```

mmshutdown command

Unmounts all GPFS file systems and stops GPFS on one or more nodes.

Synopsis

```
mmshutdown [-t UnmountTimeout] [-a | -N {Node[,Node...] | NodeFile | NodeClass}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmshutdown** command to stop the GPFS daemons on one or more nodes. If no operand is specified, GPFS is stopped only on the node from which the command was issued.

The **mmshutdown** command first attempts to unmount all GPFS file systems. If the unmount does not complete within the specified *timeout* period, the GPFS daemons shut down anyway.

Results

Upon successful completion of the **mmshutdown** command, these tasks are completed:

- GPFS file systems are unmounted.
- GPFS daemons are stopped.

Parameters

-a Stop GPFS on all nodes in a GPFS cluster.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}
Directs the **mmshutdown** command to process a set of nodes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

Options

-t *UnmountTimeout*

The maximum amount of time, in seconds, that the unmount command is given to complete. The default timeout period is equal to:

$$60 + 3 \times \textit{number of nodes}$$

If the unmount does not complete within the specified amount of time, the command times out and the GPFS daemons shut down.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmshutdown** command.

mmshutdown

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To **stop** GPFS on all nodes in the GPFS cluster, issue this command:

```
mmshutdown -a
```

The system displays information similar to:

```
Thu Mar 15 14:02:50 EDT 2012: mmshutdown: Starting force unmount of GPFS file systems
c6flc3vp2.gpfs.net: forced unmount of /gpfs/fs1
c6flc3vp1.gpfs.net: forced unmount of /gpfs/fs1
Thu Mar 15 14:03:00 EDT 2012: mmshutdown: Shutting down GPFS daemons
c6flc3vp3.gpfs.net: Shutting down!
c6flc3vp4.gpfs.net: Shutting down!
c6flc3vp4.gpfs.net: 'shutdown' command about to kill process 23649
c6flc3vp4.gpfs.net: Unloading modules from /lib/modules/2.6.27.19-5-ppc64/extra
c6flc3vp4.gpfs.net: Unloading module mmfs26
c6flc3vp1.gpfs.net: Shutting down!
c6flc3vp4.gpfs.net: Unloading module mmfslinux
c6flc3vp1.gpfs.net: 'shutdown' command about to kill process 7667944
c6flc3vp2.gpfs.net: Shutting down!
c6flc3vp2.gpfs.net: 'shutdown' command about to kill process 5701764
c6flc3vp2.gpfs.net: Master did not clean up; attempting cleanup now
c6flc3vp2.gpfs.net: Thu Mar 15 14:04:05.114 2012: mmfsd is shutting down.
c6flc3vp2.gpfs.net: Thu Mar 15 14:04:05.115 2012: Reason for shutdown: mmfsadm shutdown command timed out
c6flc3vp2.gpfs.net: Thu Mar 15 14:04:06 EDT 2012: mmcommon mmfsdown invoked. Subsystem: mmfs Status: down
c6flc3vp2.gpfs.net: Thu Mar 15 14:04:06 EDT 2012: mmcommon: Unmounting file systems ...
Thu Mar 15 14:04:10 EDT 2012: mmshutdown: Finished
```

2. To stop GPFS on only node **k164n04**, issue this command:

```
mmshutdown -N k164n04
```

The system displays information similar to:

```
Thu Mar 15 14:00:12 EDT 2012: mmshutdown: Starting force unmount of GPFS file systems
k164n04: forced unmount of /gpfs/fs1
Thu Mar 15 14:00:22 EDT 2012: mmshutdown: Shutting down GPFS daemons
k164n04: Shutting down!
k164n04: 'shutdown' command about to kill process 7274548
Thu Mar 15 14:00:45 EDT 2012: mmshutdown: Finished
```

See also

- “mmgetstate command” on page 503
- “mmlscluster command” on page 524
- “mmstartup command” on page 678

Location

```
/usr/lpp/mmfs/bin
```

mmsmb command

Administers SMB exports, export ACLs, and global configuration.

Synopsis

```

| mmsmb export list [ListofSMBExports | --raw | --option Arg | --export-regex Arg
| | --header N | --all | --key-info Arg]

or

| mmsmb export add SMBExport path [--option SMBOptionvalue | --key-info Arg]

or

| mmsmb export change SMBExport [--option SMBOptionvalue | --remove SMBOption | --key-info Arg]

or

| mmsmb export remove SMBExport [--force]

or

| mmsmb config list [ListofSMBOptions | -Y | --supported | --header N | --key-info Arg]

or

| mmsmb config change [--option SMBOptionValue | --remove SMBOption | --key-info Arg]

or

mmsmb exportacl getid ExportName { Name | --user UserName | --group GroupName
| --system SystemName }

or

mmsmb exportacl list ExportName { Name [ --viewsddl ] }

or

mmsmb exportacl add ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID } --access Access --permissions Permissions

or

mmsmb exportacl change ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID } --access Access --permissions Permissions

or

mmsmb exportacl remove ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID } --access [ Access --permissions Permissions ]

or

mmsmb exportacl replace ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID } --access Access --permissions Permissions

or

mmsmb exportacl delete ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID }

```

Note: For **mmsmbexport**, you can specify **--option --remove** multiple times but you cannot specify both of these options simultaneously. This does not apply to **mmsmb exportacl**.

mmsmb

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

The protocol functions provided in this command, or any similar command, are generally referred to as CES (Cluster Export Services). For example, protocol node and CES node are functionally equivalent terms.

Description

Use the **mmsmb** command to administer SMB exports and global configuration.

Using the **mmsmb export** command, you can do the following tasks:

- Create the specified SMB export. The **mmsmb export add** command creates the specified export for the specified path. Any supported SMB option can be specified by repeating **--option**. Also the substitution values %D for the domain, %U for session user name and %G for the primary group of %U are supported as part of the specified path. The % character is not allowed in any other context. If the export exists but the path does not exist or if it is not inside the GPFS file system, the command returns with an error. When one or more substitution variables are used, only the sub-path to the first substitution variable is checked. If authentication on the cluster is not enabled, this command will terminate with an error.
- Change the specified SMB export using the **mmsmb export change** command.
- Delete an SMB export using the **mmsmb export remove** command. Existing connections to the deleted SMB export will be disconnected. This can result in data loss for files being currently open on the affected connections.
- List the SMB exports using the **mmsmb export list** command. The command displays the configuration of options for each SMB export. If no specific options are specified, the command displays all SMB exports with the SMB options browseable; guest ok; smb encrypt as a table. Each row represents an SMB export and each column represents an SMB option.

Using the **mmsmb config** command, you can do the following tasks:

- Change, add or remove the specified SMB option for the SMB configuration. Use the **mmsmb config change** command to change the global configuration.
- List the global configuration of SMB exports. Use the **mmsmb config list** command to display the global configuration options of SMB exports. If no specific options are specified, the command displays all SMB option-value pairs.

Using the **mmsmb exportacl** command, you can do the following tasks:

- Retrieve the ID of the specified user/group/system.
- List, change, add, remove, replace and delete the ACL associated with an export.
- The add option has two mandatory arguments: **--access** and **--permissions**.

Parameters

mmsmb export

list

Lists SMB exports.

ListofSMBExports

Specifies the list of SMB exports that needs to be listed as blank separated strings.

--option arg

arg {*key* | **all** | **unsupported**}

- *key*, specifies only the supported SMB option to be listed.
- **all**, displays all used SMB options.

- **unsupported**, detects and displays all SMB exports with unsupported SMB options. The path of all unsupported options are listed for each SMB export.

| **--export-regex** *arg*

arg is a regular expression against which the exportnames are matched. Only matching exports are shown. If this option is not specified and if *ListofSMBExports* are also not specified, all existing exports are displayed.

| **--all**

Displays all defined SMB options. Similar to **--option all**.

add

Creates the specified SMB export on a GPFS file system with NFSv4 ACLs enforced. You can verify whether your GPFS files system has been configured correctly by using the **mm1sfs** command. For example, **mm1sfs gpfs0 -k**.

path

Specifies the path of the SMB export that needs to be added.

| **--option** *SMBOption=value*

Specifies the SMB option for the SMB protocol. If it is not a supported SMB option or the value is not allowable for this SMB option, the command terminates with an error. If this option is not specified, the default options: `guest ok = no` and `smb encrypt = auto` are set.

change

Modifies the specified SMB export.

| **--option** *SMBOption=value*

Specifies the SMB option for the SMB protocol. If the SMB option is not configured for the specified export, it will be added with the specified value. If the SMB option is not supported or the value is not allowable for this SMB option the command terminates with an error. If no value is specified, the specified SMB option is set to default by removing the current setting from the configuration.

| **--remove** *SMBOption*

Specifies the SMB option that is to be removed. If the SMB option is supported it will be removed from the specified export. The default value becomes active. If the SMB option is not supported, the command terminates with an error.

remove

Deletes the specified SMB export.

--force

Suppresses confirmation questions.

SMBExport

Specifies the SMB export that needs to be listed.

List of supported SMB options for the mmsmb export {list | add | change | remove} command:

admin users

Using this option, administrative users can be defined in the format of `admin users=user1;user2;..;usern`. The users must be domain users. Use of the parameter is not recommended for permanent use, files/directories created by the defined admin user will be owned by root and not by the user that had connected.

browseable

If the value is set as `yes`, the export is shown in the Windows Explorer browser when browsing the file server. By default, this option is enabled.

comment

Description of the export.

mmsmb

csc policy

`csc policy` stands for client-side caching policy, and specifies how clients that are capable of offline caching cache the files in the share. The valid values are: `manual` and `disable`. Setting `csc profile = disable` disables offline caching. For example, this can be used for shares containing roaming profiles. By default, this option is set to the value `manual`.

fileid:algorithm

This option allows to control the level of enforced data integrity. If the data integrity is ensured on the application level, it can be beneficial in cluster environments to reduce the level of enforced integrity for performance reasons.

`fsname` is the default option and ensures data integrity in the entire cluster by managing concurrent access to files and directories cluster-wide.

The `fsname_norootdir` value disables synchronization of directory locks for the root directory of the specified export, but will keep lock **fileid:algorithm** for all files and directories within and underneath the share root.

The `fsname_nodirs` value disables synchronization of directory locks across the cluster nodes, but will leave lock **fileid:algorithm** enabled for files.

The `hostname` value completely disables cross-node lock **fileid:algorithm** for both directories and files.

By default, the `fsname` value is set that enables cross-node lock **fileid:algorithm** for both directories and files.

Note: Data integrity is ensured if an application does not use multiple processes to access the data at the same time, for example, reading of file content does not happen while another process is still writing to the file. Without locking, the consistency of files is no longer guaranteed on protocol level. If data integrity is not ensured on application level this can lead to data corruption. For example, if two processes modify the same file in parallel, assuming that they have exclusive access.

gpfs:leases

gpfs:leases are cross protocol oplocks (opportunistic locks), that means an SMB client can lock a file that provides the user improved performance while reading or writing to the file because no other user read or write to this file. If the value is set as `yes`, clients accessing the file over the other protocols can break the lock of a SMB client and the user gets informed when another user is accessing the same file at the same time.

gpfs:recalls

If the value is set as `yes` files that have been migrated from disk will be recalled on access. By default, this is enabled. If `recalls = no` files will not be recalled on access and the client will receive `ACCESS_DENIED` message.

gpfs:sharemodes

An application can set share modes. If you set `gpfs:sharemodes = yes`, using the `mmsmb export` change `SMBexport --option "gpfs:sharemodes = yes"` the **sharemodes** specified by the application will be respected by all protocols and not only by the SMB protocol. If you set `gpfs:sharemodes = no` the **sharemodes** specified by the application will only be respected by the SMB protocol. For example, the NFS protocol will ignore the **sharemode** set by the application.

The application can set the following **sharemodes**: `SHARE_READ` or `SHARE_WRITE` or `SHARE_READ` and `SHARE_WRITE` or `no sharemodes`.

gpfs:syncio

If the value is set as `yes`, it specifies the files in an export, for which the setting is enabled, are opened with the `O_SYNC` flag. Accessing a file is faster if **gpfs:syncio** is set to `yes`.

Performance for certain workloads can be improved when SMB accesses the file with the `O_SYNC` flag set. For example, updating only small blocks in a large file as observed with database

applications. The underlying GPFS behavior is then changed to not read a complete block if there is only a small update to it. By default, this option is disabled.

guest ok

By default, this parameter is set to no.

hide unreadable

If the value is set as yes, all files and directories that the user has no permission to read is hidden from directory listings in the export. The `hideunreadable=yes` option is also known as access-based enumeration because when a user is listing (enumerating) the directories and files within the export, they only see the files and directories that they have read access to. By default, this option is disabled.

Note: If the value is set as yes, there is a negative impact to the read and write performance of data in the export.

oplocks

If the value is set as yes, a client may request an opportunistic lock (**oplock**) from an SMB server when it opens a file. If the server grants the request, the client can cache large chunks of the file without informing the server what it is doing with the cached chunks until the task is completed. Caching large chunks of a file saves a lot of network I/O round-trip time and enhances performance. By default, this option is enabled.

Note: While **oplocks** can enhance performance, they can also contribute to data loss in case of SMB connection breaks/timeouts. To avoid the loss of data in case of an interface node failure or storage timeout, you might want to disable **oplocks**.

Opportunistic locking allows a client to notify the SMB server that it will be the exclusive writer of the file. It also notifies the SMB server that it will cache its changes to that file on its own system and not on the SMB server to speed up file access for that client. When the SMB server is notified about a file being opportunistically locked by a client, it marks its version of the file as having an opportunistic lock and waits for the client to complete work on the file. The client has to send the final changes back to the SMB server for synchronization. If a second client requests access to that file before the first client has finished working on it, the SMB server can send an `oplock break` request to the first client. This request informs the client to stop caching its changes and return the current state of the file to the server so that the interrupting client can use it. An opportunistic lock, however, is not a replacement for a standard deny-mode lock. There are many use cases when the interrupting process to be granted an `oplock break` only to discover that the original process also has a deny-mode lock on the file.

posix locking

If the value is set as yes, it will be tested if a byte-range (`fcntl`) lock is already present on the requested portion of the file before granting a byte-range lock to an SMB client. For improved performance on SMB-only shares this option can be disabled. Disabling locking on cross-protocol shares can result in data integrity issues when clients concurrently set locks on a file via multiple protocols, for example, SMB and NFS.

read only

If the value is set as yes, files cannot be modified or created on this export independent of the ACLs. By default, the value is no.

smb encrypt

This option controls whether the remote client is allowed or required to use SMB encryption. Possible values are `auto`, `mandatory`, and `disabled`. This is set when the export is created with default value. Clients may chose to encrypt the entire session, not just traffic to a specific export. The server would return access denied message to all non-encrypted requests on such an export. Selecting encrypted traffic reduces throughput as smaller packet sizes must be used as well as the overhead of encrypting and signing all the data. If SMB encryption is selected, Windows style SMB signing is no longer necessary, as the GSSAPI flags use select both signing

mmsmb

and sealing of the data. When set to `auto`, SMB encryption is offered, but not enforced. When set to `mandatory`, SMB encryption is required and if set to `disabled`, SMB encryption can not be negotiated.

syncops:oncloses

This option ensures that the file system synchronizes data to the disk each time a file is closed after writing. The written data is flushed to the disk. By default, this option is enabled.

Note: Disabling this option increases the risk of data loss in case of a node failure.

mmsmb config

list

Lists the global configuration options of SMB exports.

ListofSMBOptions

Specifies the list of SMB options that needs to be listed.

--supported

Displays all changeable SMB options and their values.

change

Modifies the global configuration options of SMB exports.

--option SMBOption=value

Sets the value of the specified SMB option. If no value is given, the SMB option is removed.

Specifies the SMB option for the SMB protocol. If the SMB option is not configured for global configuration, it will be added with the specified value. If no value is specified, the specified SMB option is set to default and removed from the global configuration. If the SMB option is not supported or the value is not allowable for the SMB option, the command terminates with an error.

--remove SMBOption

Specifies the SMB option that is to be removed. If the SMB option is supported it will be removed from the global configuration. The default value becomes active. If the SMB option is not supported, the command terminates with an error.

List of supported SMB options by the `mmsmb config {list | change}` command:

gpfs:dfreequota

gpfs:dfreequota stands for disk Free Quota. If the value is set to `yes` the free space and size reported to a SMB client for a share will be adjusted according to the applicable quotas. The applicable quotas are the quota of the user requesting this information, the quota of the user's primary group and the quota of the fileset containing the export.

restrict anonymous

The setting of this parameter determines whether access to information is allowed or restricted for anonymous users. The options are:

`restrict anonymous = 2`: anonymous users are restricted from accessing information.

`restrict anonymous = 0`: anonymous users are allowed to access information. This is the default setting.

`restrict anonymous = 1`: is not supported

server string

server string stands for Server Description. It specifies the server description for SMB protocol. Server description with special characters must be provided in single quotes.

mmsmb exportacl

getid

Retrieve the ID of the specified user/group/system.

```
mmsmb exportacl getid --user myUser
mmsmb exportacl getid --group myGroup
mmsmb exportacl getid --system mySystem
```

list

List can only take viewing options.

```
mmsmb exportacl list myExport          Show the export ACL for this exportname
mmsmb exportacl list                  Show all the export ACLs
mmsmb exportacl list myExport --viewsddl Show the export ACL for this exportname in sddl format.
```

add

Add will add a new permission to the export ACL. It will include adding a user, group or system. The options are:

- {User/group/system name} If you do not specify the type of name, the system will prioritize in this order:
- --user
- --group
- --system, or
- --SID (this can be the SID for a user, group or system).

Mandatory arguments are:

- --access: ALLOWED or DENIED
- --permissions: One of FULL, CHANGE, or READ or any combination of RWXDPO.

Examples:

```
mmsmb exportacl add myExport0 myUser --access ALLOWED --permissions FULL
mmsmb exportacl add myExport1 --user user01 --access ALLOWED --permissions RWO
mmsmb exportacl add myExport2 --group group01 --access DENIED --permissions RWXDP
```

change

Change will update the specified ACE in an export ACL. The options are:

- {User/group/system name} If you do not specify the type of name, the system will prioritize in this order:
- --user
- --group
- --system
- --SID (This can be the SID for a user, group or system).

Mandatory arguments are:

- --access: ALLOWED or DENIED
- --permissions: One of FULL, CHANGE, or READ or any combination of RWXDPO.

Examples:

```
mmsmb exportacl change myExport --user myUser --access ALLOWED --permissions RWX
mmsmb exportacl change myExport --group allUsers --access ALLOWED --permissions R
```

remove

Remove will remove the ACE for the specified user/group/system from the ACL.

The user, group or system will be removed automatically for a specified name.

In the event that the system is unable to locate an ACE within the export ACL that it can remove the permissions for as instructed, an error will be issued to the user informing them of this.

The options are:

mmsmb

- {User/group/system name} If you do not specify the type of name, the system will prioritize in this order:
- --user
- --group
- --system
- --SID (This can be the SID for a user, group or system).

Optional arguments are:

- --access: ALLOWED or DENIED
- --permissions: One of FULL, CHANGE, or READ or any combination of RWXDPO.

Examples:

```
mmsmb exportacl remove myExport01 UserName --access ALLOWED --permissions CHANGE
mmsmb exportacl remove myExport02 GroupName --access DENY --permissions READ
mmsmb exportacl remove myExport03 UserName
```

replace

The replace command replaces all the permissions in a export ACL with those indicated in its ACE specification. It is therefore a potentially destructive command and will include a confirmation. This confirmation can be overridden with the --force command. The options are:

- {User/group/system name} If you do not specify the type of name, the system will prioritize in this order:
- --user
- --group
- --system
- --SID (This can be the SID for a user, group or system)
- --force.

Mandatory arguments are:

- --access: ALLOWED or DENIED
- --permissions: One of FULL, CHANGE, or READ or any combination of RWXDPO.

Examples:

```
mmsmb exportacl replace myExport01 --user user01 --access ALLOWED --permissions FULL
mmsmb exportacl replace myExport02 --group group01 --access ALLOWED --permissions READ --force
```

delete

The Delete command will remove an enter export ACL. It therefore does not require a system, id or user identified as they are not appropriate in this case. All it needs is the name of an export for which the export ACL will be deleted.

Delete will include a confirmation. This can be overridden using the --force parameter.

Examples:

```
mmsmb exportacl delete myExport01
mmsmb exportacl delete myExport02 --force
```

Parameters common for both mmsmb export and mmsmb config commands

--raw

Displays command output in machine-readable format.

Note: The output includes colon ":" as a field separator. If the values in the output include a ":", then it is replaced by "%3A". If the values include a "%". then it is replaced by "%25".

--header *n*

Repeats the output table header every *n* lines for a table that is spread over multiple pages. The value *n* can be of any integer value.

- | **--key-info** *arg*
Displays the supported SMB options and their possible values.
- | *arg* *SMBoption* | supported
SMBoption
Specifies the SMB option.
- | **supported**
Displays descriptions for all the supported SMB options.

Exit status

- 0 Successful completion.
- nonzero A failure has occurred.

Security

You must have root authority to run the **mmsmb** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

mmsmb config list

1. Show descriptions of all the supported SMB configuration options.

```
mmsmb config list --key-info supported
```

The system displays output similar to this:

```
Supported smb options with allowed values:
gpfs:dfreequota    = yes, no
restrict anonymous = 0, 2
server string      = any
```

2. List the SMB option that specifies whether anonymous access is allowed or not.

```
mmsmb config list "gpfs:dfreequota"
```

The system displays output similar to this:

```
SMB option      value
gpfs:dfreequota yes
```

3. Display the SMB configuration options in a machine-readable format.

```
mmsmb config list -Y
```

The system displays output similar to this:

```
add share command:aio read size:aio write size:aio_pthread%3Aaio open:async smb echo handler:auth methods:change
notify:change share command:client NTLMv2 auth:ctdb locktime warn threshold:debug hires timestamp:delete share
command:dfree cache time:disable netbios:disable spoolss:dmap support:ea support:fileid%3Amapping:force unknown
acl user:gencache%3Astabilize_count:gpfs%3Adfreequota:gpfs%3Ahsmb:gpfs%3Aleases:gpfs%3Aprealloc:gpfs%3Asharemodes:
```

mmsmb config change

1. Show descriptions of all the supported SMB configuration options.

```
| mmsmb config change --key-info supported
```

The system displays output similar to this:

mmsmb

Supported smb options with allowed values:

```
gpfs:dfreequota    = yes, no
restrict anonymous  = 0, 2
server string      = any
```

2. Change an SMB configuration option.

```
mmsmb config change --option "restrict anonymous=2"
```

You can confirm the change by using this **mmsmb config list "restrict anonymous"** command.

3. Remove an SMB configuration option.

```
mmsmb config change --remove "server string"
```

The system displays output similar to this:

Warning:

```
Unused options suppressed in display:
server string
```

mmsmb export list

1. To list all SMB options for export myExport, issue this command:

```
mmsmb export list --export myExport --option all
```

The system displays output similar to this:

```
export    path                smb encrypt  oplocks    guest ok
myExport  /mnt/gpfs0/shared          auto         yes        no
```

2. To list SMB option csc policy for SMB export myexport and all SMB exports starting with foo followed by any quantity of 1 and ending on 2 or 3, issue this command:

```
mmsmb export list --option "csc policy" myexport --export-regex "foo1*[23]$"
```

The system displays output similar to this:

```
export    path                                csc policy
myExport  /mnt/gpfs0/shared                      - - -
foo11b23  /mnt/gpfs0/testExport/testSmbEncrypt  disable
foo11b23  /mnt/gpfs0/testExport/testSmbEncrypt  documents
```

3. To list all exports where unsupported SMB options are set, issue this command:

```
mmsmb export list --option unsupported
```

The system displays output similar to this:

```
export          path                mangled names
hasForbiddenOptions  /mnt/gpfs0/shared          yes
```

mmsmb export add

1. To create an export myExport as default SMB export for path /ibm/gpfs0/myFolder with default options, issue this command:

```
mmsmb export add myExport "/ibm/gpfs0/myFolder"
```

The system displays output similar to this:

```
mmsmb export add: The SMB export was created successfully.
```

mmsmb export change

1. To change the SMB option oplocks to value yes for SMB export myExport, issue this command:

```
mmsmb export change myExport --option "oplocks"="yes"
```

You can confirm the change by using this **mmsmb export list --option "oplocks" myExport** command. The system displays output similar to this:

```
export    path                oplocks
myExport  /mnt/gpfs0/shared          yes
```

2. To remove the SMB option `oplocks` from SMB export `myExport`, issue the following commands:
`mmsmb export change myExport --remove "oplocks"`

You can confirm the change by using this `mmsmb export list --option all myExport` command.

mmsmb export remove

1. To delete the SMB export `myExport` with confirmation, issue this command:
`mmsmb export remove myExport`

The system asks for confirmation, similar to this:

Do you really want to perform the operation (yes/no - default no):

2. To delete the SMB export `myExport` without confirmation:
`mmsmb export remove myExport --force`

The system removes the SMB export without any confirmation.

See also

- “`mmnfs` command” on page 570
- “`mmces` command” on page 304
- “`mmlsfs` command” on page 536

Location

`/usr/lpp/mmfs/bin`

mmsnapdir command

Controls how the special directories that connect to snapshots appear.

Synopsis

```
mmsnapdir Device [-r | -a]
    [--show-global-snapshots {rootfileset | allfilesets}]
    [{[--fileset-snapdir FilesetSnapDirName] [--global-snapdir GlobalSnapDirName]}
    | [--snapdir | -s} SnapDirName}]
```

or

```
mmsnapdir Device [-q]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmsnapdir** command to control how the special directories that connect to snapshots appear. Both the name of the directories and where they are accessible can be changed.

Global snapshots appear in a subdirectory in the root directory of the file system, whose default name is **.snapshots**. Fileset snapshots appear in a similar **.snapshots** subdirectory located in the root directory of each independent fileset. These special subdirectories are collectively referred to as *snapdirs*. Note that the root directory of the file system and the root directory of the root fileset are the same, so global snapshots and fileset snapshots of the root fileset will both appear in the same *snapdir*.

If you prefer to access the snapshots from each directory rather than traversing through the root directory, you can use an invisible directory to make the connection by issuing the **mmsnapdir** command with the **-a** option (see “Examples”). The **-a** option enables an invisible directory in each directory in the active file system (they do not appear in directories in snapshots) that contains a subdirectory for each existing snapshot of the file system (in the root fileset) or fileset (in other independent filesets). These subdirectories correspond to the copy of the active directory in the snapshot with the same name. For example, if you enter **ls -a /fs1/userA**, the (invisible) **.snapshots** directory is not listed. However, you can use **ls /fs1/userA/.snapshots**, for example, to confirm that **.snapshots** is present and contains the snapshots holding copies of **userA**. When the **-a** option is enabled, the paths **/fs1/.snapshots/Snap17/userA** and **/fs1/userA/.snapshots/Snap17** refer to the same directory, namely **userA** at the time when **Snap17** was created. The **-r** option (root-directories-only), which is the default, reverses the effect of the **-a** option (all-directories), and disables access to snapshots via *snapdirs* in non-root directories.

If you prefer to access global snapshots from the root directory of all independent filesets, use the **mmsnapdir** command with the **--show-global-snapshots allfilesets** option. With this option, global snapshots will also appear in the *snapdir* in the fileset root directory. The global snapshots will also appear in the *snapdirs* in each non-root directory if all-directories (the **-a** option) is enabled. To return to the default setting use **--show-global-snapshots rootfileset**, and global snapshots will only be available in root of the file system, or the root fileset, if all-directories is enabled.

The name of the *snapdir* directories can be changed using the **--snapdir** (or **-s**) option. This name is used for both global and fileset snapshots in both fileset root directories and, if all-directories is enabled, non-root directories also. The *snapdir* name for global and fileset snapshots can be specified separately using the **--global-snapdir** and **--fileset-snapdir** options. If these names are different, two *snapdirs* will appear in the file system root directory, with the global and fileset snapshots listed separately. When **--show-global-snapshots** is set to **allfilesets**, two *snapdirs* will appear in fileset root directories also, and when all-directories (the **-a** option) is specified, the two *snapdirs* will be available in non-root directories as well. If **--global-snapdir** is specified by itself, the fileset *snapdir* name is left unchanged, and vice versa

if **--fileset-snapdir** option is used. Setting both snapdirs to the same name is equivalent to using the **--snapdir** option. The snapdir name enabled in non-root directories by all-directories is always the same as the name used in root directories.

For more information on global snapshots, see *Creating and maintaining snapshots of GPFS file systems* in the *IBM Spectrum Scale: Advanced Administration Guide*.

For more information on fileset snapshots, see *Fileset-level snapshots* in the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system. File system names need not be fully-qualified. **fs0** is just as acceptable as **/dev/fs0**.

This must be the first parameter.

- a** Adds a snapshots subdirectory to all subdirectories in the file system.
- r** Reverses the effect of the **-a** option. All invisible snapshot directories are no longer accessible. The snapshot directory under the file system root directory is not affected.

--show-global-snapshots {rootfileset | allfilesets}

This option controls whether global snapshots are accessible through a subdirectory under the root directory of all independent filesets (**allfilesets**) or only in the file system root (**rootfileset**). For example, issuing the following command:

```
mmsnapdir fs1 --show-global-snapshots allfilesets
```

specifies that the root directory of each independent fileset will contain a **.gsnaps** subdirectory listing all global snapshots, such as **/fs1/junctions/FsetA/.gsnaps** and **/fs1/junctions/FsetA/.fsnaps**. This can be used to make global snapshots accessible to clients, for example NFS users, that do not have access to the file system root directory.

Specifying **rootfileset** reverses this feature, restoring the default condition, so that global snapshots are only visible in the root fileset.

--fileset-snapdir *FilesetSnapDirName*

--global-snapdir *GlobalSnapDirName*

The **--global-snapdir** option specifies the name for the directory where global snapshots are listed.

The **--fileset-snapdir** option specifies the name for the directory where fileset snapshots are listed.

These options can be specified together or separately, in which case only the corresponding snapdir is changed. Neither option may be specified with **--snapdir**, which sets both to the same name.

For example, after issuing the command:

```
mmsnapdir fs1 --fileset-snapdir .fsnaps --global-snapdir .gsnaps
```

the directory **/fs1/.gsnaps** will list all global snapshots and **/fs1/.fsnaps** will only list fileset snapshots of the root fileset. Fileset snapshots of other independent filesets will be listed in **.fsnaps** under the root directory of each independent fileset, such as **/fs1/junctions/FsetA/.fsnaps**.

--snapdir | **-s** *SnapDirName*

Changes the name of the directory for both global and fileset snapshots to *SnapDirName*. This affects both the directory in the file system root as well as the invisible directory in the other file system directories if the **-a** option has been enabled. The root and non-root snapdirs cannot be given different names.

- q** Displays current snapshot settings. The **-q** option cannot be specified with any other options. This is the default if no other options are specified.

mmsnapdir

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

If you are a root user, the node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See “Requirements for administering a GPFS file system” on page 1.

You must be a root user to use all of the **mmsnapdir** options. Non-root users can only use the **-q** option.

If you are a non-root user, you may only specify file systems that belong to the same cluster as the node on which the **mmsnapdir** command was issued.

Examples

1. To rename the **.snapshots** directory (the default snapshots directory name) to **.link** for file system **fs1**, issue the command:

```
mmsnapdir fs1 -s .link
```

After the command has been issued, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3

/fs1/.link/snap1/file1
/fs1/.link/snap1/userA/file2
/fs1/.link/snap1/userA/file3
```

2. To add the **.link** subdirectory to all subdirectories in the file system, issue:

```
mmsnapdir fs1 -a
```

After the command has been issued, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3
/fs1/userA/.link/snap1/file2
/fs1/userA/.link/snap1/file3

/fs1/.link/snap1/file1
/fs1/.link/snap1/userA/file2
/fs1/.link/snap1/userA/file3
```

The **.link** subdirectory under the root directory and under each subdirectory of the tree provides two different paths to each snapshot copy of a file. For example, **/fs1/userA/.link/snap1/file2** and **/fs1/.link/snap1/userA/file2** are two different paths that access the same snapshot copy of **/fs1/userA/file2**.

3. To reverse the effect of the previous command, issue:

```
mmsnapdir fs1 -r
```

After the command has been issued, the directory structure would appear similar to:

```
/fs1/file1
/fs1/userA/file2
/fs1/userA/file3

/fs1/.link/snap1/file1
/fs1/.link/snap1/userA/file2
/fs1/.link/snap1/userA/file3
```

4. To display the current snapshot settings, issue:

```
mmsnapdir fs1 -q
```

The system displays output similar to:

```
Snapshot directory for "fs1" is ".link" (root directory only)
```

If there are independent filesets, fileset snapshots, or the global and fileset snapshot directory names are different in the file system, the system displays output similar to:

```
Fileset snapshot directory for "fs1" is ".link" (root directory only)
```

```
Global snapshot directory is ".link" in root directory only
```

See also

- “mmcrsnapshot command” on page 431
- “mmdelsnapshot command” on page 467
- “mmlssnapshot command” on page 563
- “mmrestorefs command” on page 635

Location

```
/usr/lpp/mmfs/bin
```

mmstartup

mmstartup command

Starts the GPFS subsystem on one or more nodes.

Synopsis

```
mmstartup [-a | -N {Node[,Node...] | NodeFile | NodeClass}] [-E EnvVar=value ...]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmstartup** command to start the GPFS daemons on one or more nodes. If no operand is specified, GPFS is started only on the node from which the command was issued.

Parameters

-a Start GPFS on all nodes in a GPFS cluster.

-N {Node[,Node...] | NodeFile | NodeClass}

Directs the **mmstartup** command to process a set of nodes.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

-E EnvVar=value

Specifies the name and value of an environment variable to be passed to the GPFS daemon. You can specify multiple **-E** options.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmstartup** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To start GPFS on all nodes in the GPFS cluster, issue this command:

```
mmstartup -a
```

The system displays information similar to:

```
Thu Aug 12 13:22:40 EDT 2004: 6027-1642 mmstartup: Starting GPFS ...
```

See also

- “mmgetstate command” on page 503
- “mmlscluster command” on page 524
- “mmshutdown command” on page 661

Location

/usr/lpp/mmfs/bin

mmtracectl command

Sets up and enables GPFS tracing.

Synopsis

```
mmtracectl { --start | --stop | --off | --set }
            [--trace={io | all | def | "Class Level [Class Level ...]" }]
            [--trace-recycle={off | local | global | globalOnShutdown }]
            [--aix-trace-buffer-size=BufferSize]
            [--tracedev-buffer-size=BufferSize]
            [--trace-file-size=FileSize] [--trace-dispatch={yes | no }]
            [--tracedev-compression-level=Level]
            [--tracedev-write-mode={blocking | overwrite }]
            [--tracedev-timeformat={relative | absolute | calendar }]
            [--tracedev-overwrite-buffer-size=Size]
            [--format | --noformat]
            [-N {Node [,Node...] | NodeFile | NodeClass }]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Attention: Use this command only under the direction of the IBM Support Center.

Use the **mmtracectl** command to perform the following functions:

- Start or stop tracing.
- Turn tracing on (start or set trace recycle) or off on the next session. This is a persistent setting to automatically start trace each time GPFS starts.
- Allow for predefined trace levels: **io**, **all**, and **def**, as well as user-specified trace levels.
- Allow for changing the size of trace buffer sizes for AIX and all others using the **tracedev** option.
- Trace recycle functions, which allow for never cycling traces (**off** option), cycling traces on all nodes when GPFS ends abnormally (**global** option), and cycling traces any time GPFS goes down on all nodes (**globalOnShutdown** option).
- For Linux nodes only, this command allows you to change:
 - The trace writing mode
 - The raw data compression level

Note: Tracing on Windows requires support programs provided by Microsoft. For details about this prerequisite, see the section about configuring Windows and installing tracing support programs in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Results

GPFS tracing can be started, stopped, or related configuration options can be set.

Parameters

--start | --stop | --off | --set

Specifies the actions that the **mmtracectl** command performs, where:

start

Starts the trace.

stop

Stops the trace.

off

Clears all of the setting variables and stops the trace.

set

Sets the trace variables.

--trace={io | all | def | "Class Level [Class Level ...]"}

Allows for predefined and user-specified trace levels, where:

io Indicates trace-level settings tailored for input and output (I/O).

all

Sets trace levels to their highest setting (9).

def

Indicates that the default trace settings will be used.

"Class Level [Class Level ...]"

Specifies a trace class and level.

--trace-recycle={off | local | global | globalOnShutdown}

Controls trace recycling during daemon termination. The following values are recognized:

off

Does not recycle traces. This is the default value until **mmtracectl --start** is run. If no trace-recycle value has been explicitly set when **mmtracectl --start** is run, see the following description for **local**. The **mmtracectl --off** command will remove any explicit value for **trace-recycle**, thus effectively setting **trace-recycle** back to the **off** value.

local

Recycles traces on the local node when **mmfsd** goes down abnormally. This setting also starts traces automatically any time that **mmstartup** is run to start the GPFS daemon, which could include an autoload on a reboot. If there is no **trace-recycle** value explicitly set at the time that **mmtracectl --start** is run, **mmchconfig** will be run to explicitly set the **trace-recycle** value to **local**. The starting of the actual tracing will be delayed while that configuration change is being made.

global

Recycles traces on all nodes in the cluster when an abnormal daemon shutdown occurs.

globalOnShutdown

Recycles traces on all nodes in the cluster for normal and abnormal daemon shutdowns.

--aix-trace-buffer-size=BufferSize

Controls the size of the trace buffer in memory for AIX.

--tracedev-buffer-size=BufferSize

Specifies the trace buffer size for Linux trace in blocking mode. If **--tracedev-write-mode** is set to blocking, this parameter will be used. It should be no less than 4K and no more than 64M. The default is 4M.

Note: This option applies only to Linux nodes.

--trace-file-size=FileSize

Controls the size of the trace file. The default is 128M on Linux and 64M on other platforms.

--trace-dispatch={yes | no}

Enables AIX thread dispatching trace hooks.

--tracedev-compression-level=Level

Specifies the trace raw data compression level. Valid values are 0 to 9. A value of zero indicates no compression. A value of 9 provides the highest compression ratio, but at a lower speed. The default is 6.

Note: This option applies only to Linux nodes.

mmtracectl

--tracedev-write-mode={blocking | overwrite}

Specifies when to overwrite the old data, where:

blocking

Specifies that if the trace buffer is full, wait until the trace data is written to the local disk and the buffer becomes available again to overwrite the old data. This is the default.

overwrite

Specifies that if the trace buffer is full, overwrite the old data.

Note: This option applies only to Linux nodes.

--tracedev-timeformat={relative | absolute | calendar}

Controls time formatting in the trace records. The following values are accepted:

relative

Displays the trace time stamp in relative format, showing the number of seconds from the beginning time stamp. This is the default.

absolute

Displays the trace time stamp in absolute format, showing the number of seconds since 1/1/1970.

calendar

Displays the trace time stamp in local calendar format, showing day of the week, month, day, hours, minutes, seconds, and year.

--tracedev-overwrite-buffer-size=Size

Specifies the trace buffer size for Linux trace in overwrite mode. If **--tracedev-write-mode** is set to **overwrite**, this parameter will be used. It should be no less than 16M. The default is 64M.

Note: This option applies only to Linux nodes.

--format | --noformat

Enables or disables formatting.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes that will participate in the tracing of the file system. This option supports all defined node classes (with the exception of **mount**). The default value is **all**.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmtracectl** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

To set trace levels to the defined group of **def** and have traces start on all nodes when GPFS comes up, issue this command:


```
mmtracectl --set --trace=def --trace-recycle=global
```

The system displays output similar to:

```
mmchconfig: Command successfully completed  
mmchconfig: Propagating the cluster configuration data to all  
affected nodes. This is an asynchronous process.
```

To confirm the change, issue this command:

```
mmisconfig|grep trace,traceRecycle
```

The system displays output similar to:

```
trace all 4 tm 2 thread 1 mutex 1 vnode 2 ksvfs 3 klockl 2 io 3 pgallo 1 mb 1 lock 2 fsck 3  
traceRecycle global
```

To manually start traces on all nodes, issue this command:

```
mmtracectl --start
```

See also

- “mmchconfig command” on page 331

See the **mmtrace** shell script.

Location

```
/usr/lpp/mmfs/bin
```

mmumount command

Unmounts GPFS file systems on one or more nodes in the cluster.

Synopsis

```
mmumount {Device | MountPoint | DriveLetter |  
           all | all_local | all_remote | {-F DeviceFileName}}  
           [-f] [-a | -N {Node[,Node...]} | NodeFile | NodeClass}]
```

or

```
mmumount Device -f -C {all_remote | ClusterName} [-N {Node[,Node...]}]
```

Availability

Available on all IBM Spectrum Scale editions.

Description

Another name for the **mmumount** command is the **mmunmount** command. Either name can be used.

The **mmumount** command unmounts a previously mounted GPFS file system on one or more nodes in the cluster. If no nodes are specified, the file systems are unmounted only on the node from which the command was issued. The file system can be specified using its device name or the mount point where it is currently mounted.

Use the first form of the command to unmount file systems on nodes that belong to the local cluster.

Use the second form of the command with the **-C** option when it is necessary to force an unmount of file systems that are owned by the local cluster, but are mounted on nodes that belong to another cluster.

When a file system is unmounted by force with the second form of the **mmumount** command, the affected nodes may still show the file system as mounted, but the data will not be accessible. It is the responsibility of the system administrator to clear the mount state by issuing the **umount** command.

When multiple nodes are affected and the unmount target is identified via a mount point or a Windows drive letter, the mount point is resolved on each of the target nodes. Depending on how the file systems were mounted, this may result in different file systems being unmounted on different nodes. When in doubt, always identify the target file system with its device name.

Parameters

Device | *MountPoint* | *DriveLetter* | **all** | **all_local** | **all_remote** | {-F *DeviceFileName*}

Indicates the file system or file systems to be unmounted.

Device

Is the device name of the file system to be unmounted. File system names do not need to be fully qualified. **fs0** is as acceptable as **/dev/fs0**.

MountPoint

Is the location where the GPFS file system to be unmounted is currently mounted.

DriveLetter

Identifies a file system by its Windows drive letter.

all

Indicates all file systems that are known to this cluster.

all_local

Indicates all file systems that are owned by this cluster.

all_remote

Indicates all file systems that are owned by another cluster to which this cluster has access.

-F DeviceFileName

Specifies a file containing the device names, one per line, of the file systems to be unmounted.

This must be the first parameter.

Options

-a Unmounts the file system on all nodes in the GPFS cluster.

-f Forces the unmount to take place even though the file system may be still in use.

Use this flag with *extreme caution*. Using this flag may cause outstanding write operations to be lost. Because of this, forcing an unmount can cause data integrity failures and should be used with caution.

The **mmumount** command relies on the native **umount** command to carry out the unmount operation. The semantics of forced unmount are platform-specific. On some platforms (such as Linux), even when forced unmount is requested, a file system cannot be unmounted if it is still referenced by the system kernel. Examples of such cases are:

- Open files are present in the file system
- A process uses a subdirectory in the file system as the current working directory
- The file system is NFS-exported

To unmount a file system successfully in such a case, it may be necessary to identify and stop the processes that are referencing the file system. System utilities like **lsof** and **fuser** could be used for this purpose.

-C {all_remote | ClusterName}

Specifies the cluster on which the file system is to be unmounted by force. **all_remote** denotes all clusters other than the one from which the command was issued.

-N {Node[,Node...] | NodeFile | NodeClass}

Specifies the nodes on which the file system is to be unmounted.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

This command does not support a *NodeClass* of **mount**.

When the **-N** option is specified in conjunction with **-C ClusterName**, the specified node names are assumed to refer to nodes that belong to the specified remote cluster (as identified by the **mmlsmount** command). The **mmumount** command cannot verify the accuracy of this information. *NodeClass* and *NodeFile* are not supported in conjunction with the **-C** option.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmumount** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmumount

Examples

1. To unmount file system **fs1** on all nodes in the cluster, issue this command:

```
mmumount fs1 -a
```

The system displays output similar to:

```
Fri Feb 10 15:51:25 EST 2006: mmumount: Unmounting file systems ...
```

2. To force unmount file system **fs2** on the local node, issue this command:

```
mmumount fs2 -f
```

The system displays output similar to:

```
Fri Feb 10 15:52:20 EST 2006: mmumount: Unmounting file systems ...  
forced unmount of /fs2
```

See also

- “mmmumount command” on page 568
- “mmlsmount command” on page 544

Location

```
/usr/lpp/mmfs/bin
```

mmunlinkfileset command

Removes the junction to a GPFS fileset.

Synopsis

```
mmunlinkfileset Device {FilesetName | -J JunctionPath} [-f]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

The **mmunlinkfileset** command removes the junction to the fileset. The junction can be specified by path or by naming the fileset that is its target. The unlink fails if there are files open in the fileset, unless the **-f** flag is specified. The root fileset may not be unlinked.

Attention: If you are using the TSM Backup-Archive client, use caution when you unlink filesets that contain data backed up by TSM. TSM tracks files by pathname and does not track filesets. As a result, when you unlink a fileset, it appears to TSM that you deleted the contents of the fileset. Therefore, the TSM Backup-Archive client inactivates the data on the TSM server which may result in the loss of backup data during the expiration process.

For information on GPFS filesets, see the *IBM Spectrum Scale: Advanced Administration Guide*.

Parameters

Device

The device name of the file system that contains the fileset.

File system names need not be fully-qualified. **fs0** is as acceptable as **/dev/fs0**.

FilesetName

Specifies the name of the fileset to be removed.

-J *JunctionPath*

Specifies the name of the junction to be removed.

A junction is a special directory entry that connects a name in a directory of one fileset to the root directory of another fileset.

-f Forces the unlink to take place even though there may be open files. This option forcibly closes any open files, causing an **errno** of **ESTALE** on their next use of the file.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmunlinkfileset** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

mmunlinkfileset

Examples

1. This command indicates the current configuration of filesets for file system **gpfs1**:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status    Path
root           Linked   /gpfs1
fset1          Linked   /gpfs1/fset
```

This command unlinks fileset **fset1** from file system **gpfs1**:

```
mmunlinkfileset gpfs1 fset1
```

The system displays output similar to:

```
Fileset 'fset1' unlinked.
```

To confirm the change, issue this command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status    Path
root           Linked   /gpfs1
fset1          Unlinked --
```

2. This command indicates the current configuration of filesets for file system **gpfs1**:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status    Path
root           Linked   /gpfs1
fset1          Linked   /gpfs1/fset1
```

This command unlinks junction path **/gpfs1/fset1** from file system **gpfs1**:

```
mmunlinkfileset gpfs1 -J /gpfs1/fset1
```

The system displays output similar to:

```
Fileset 'fset1' unlinked.
```

To confirm the change, issue this command:

```
mmlsfileset gpfs1
```

The system displays output similar to:

```
Filesets in file system 'gpfs1':
Name           Status    Path
root           Linked   /gpfs1
fset1          Unlinked --
```

See also

- “mmchfileset command” on page 365
- “mmcrfileset command” on page 408
- “mmdelfileset command” on page 455
- “mmlinkfileset command” on page 517
- “mmlsfileset command” on page 532

Location

/usr/lpp/mmfs/bin

mmuserauth command

Manages the authentication of protocol users who need to access the protocol data that is stored on the system. You can create, list, verify, and remove authentication configuration using this command.

Synopsis

```
mmuserauth service create --data-access-method{file|object}
  --type {ldap|local|ad|nis|userdefined}
  --servers[IP address/hostname]
  [--base-dn]
  {[--enable-anonymous-bind] | [--user-name] [--password]}
  [--enable-server-tls] [--enable-ks-ssl]
  [--enable-kerberos] [--enable-nfs-kerberos] [--enable-ks-casigning]
  [--user-dn] [--group-dn] [--netgroup-dn]
  [--netbios-name] [--domain]
  [--idmap-role{master|subordinate}] [--idmap-range] [--idmap-range-size]
  [--user-objectclass] [--group-objectclass] [--user-name-attrib]
  [--user-id-attrib] [--user-mail-attrib] [--user-filter]
  [--ks-dns-name] [--ks-admin-user] [--ks-admin-pwd]
  [--ks-swift-user] [--ks-swift-pwd] [--ks-ext-endpoint]
  [--kerberos-server] [--kerberos-realm]
  [--unixmap-domains] [ldapmap-domains]
```

Or

```
mmuserauth service list [--data-access-method {file|object|all}] [-Y]
```

Or

```
mmuserauth service check [--data-access-method {file|object|all}] [-r|--rectify]
  [-N|--nodes {node-list|cesNodes}] [--server-reachability]
```

Or

```
mmuserauth service remove --data-access-method {file|object|all} [--idmapdelete]
```

Availability

Available with IBM Spectrum Scale Standard Edition or higher.

Description

Use the **mmuserauth** commands to create and manage IBM Spectrum Scale protocol authentication and ID mappings.

Parameters

service

Manages the authentication configuration for protocol users with one of the following actions:

create

Configures authentication for object and file protocols. The authentication method for file and object cannot be configured together. The **mmuserauth service create** command needs to be submitted separately for configuring authentication for the file and object access.

list

Displays the details of the authentication method that is configured for both file and object access on the protocol nodes.

check

Verifies and corrects the authentication method that is configured for file and object access on the protocol nodes. Also checks for the existence of SSL and TLS certificates.

remove

Removes the authentication and ID maps. If you need to remove both authentication and ID maps, remove authentication first and then ID maps. That is, at first you need to run the **mmuserauth service remove** command without the **--idmapdelete** option to remove the authentication method and then run the same command with the **--idmapdelete** option to remove ID maps.

Deleting authentication and ID maps result in loss of access to data.

--data-access-method *{file|object}*

Specifies the data access method for which the authentication needs to be configured. The IBM Spectrum Scale system supports protocols such as SMB, NFS, and Object to access data that is stored in the system.

The file data access method is meant for authorizing the users who access data over SMB and NFS protocols.

--type *{ldap|local|ad|nis|userdefined}*

Specifies the type of authentication server to be integrated for file and object authentication.

ldap - Uses an external LDAP as the authentication server. This authentication type is valid for both file and object.

ad - Uses an external Microsoft Active Directory as the authentication server. This authentication type is valid for both file and object.

local - Uses an internal database for authenticating object users.

nis - Uses an NIS server as the authentication method for NFS data access. This authentication type is only used for file access.

userdefined - Uses user-defined authentication method for data access. This authentication type is valid for both file and object.

--servers *[AuthServer1[:Port],AuthServer2[:Port],AuthServer3[:Port] ...]*

Specifies the host name or IP address of the authentication server that is used for file and object. This option is only valid with **--type** {ldap|ad|nis}.

With **--type** ldap, the input value format is "serverName/serverIP:[port]". The port is optional. Default port is 389. For example, **--servers** ldapsrvr.mydomain.com:1389. Multiple LDAP servers can be specified if the value of **--data-access-method** is file. For object, only one server is considered as the authentication server at a time. Even if you specify multiple servers, only the first server in the list is considered as the authentication server.

With **--type** ad, the input value format is "serverName/serverIP". For example, **--servers** ldapsrvr.mydomain.com. For object, only one server is considered as the authentication server at a time. Even if you specify multiple servers, only the first server in the list is considered as the authentication server. Specifying multiple servers is not valid for file authentication.

With **--type** nis, the input value format is "serverName/serverIP". For example, **--servers** ldapsrvr.mydomain.com. At least one of the servers specified with **--servers** must be available while configuring authentication. This is essential for the NIS domain verification, where the availability of either 'passwd.byname' or 'netgroup' map is validated.

--base-dn *ldapBase*

Specifies the LDAP base DN of the authentication server. This option is only valid with **--type** {ldap|ad} for **--data-access-method** object and **--type** ldap for **--data-access-method** file.

--enable-anonymous-bind

Enables anonymous binding with authentication server for operations. This option is only valid with **--type** {ldap|ad} and **--data-access-method** {object}. This option is mutually exclusive with **--user-name** and **--password**.

mmuserauth

--user-name *userName*

Specifies the user name to be used to perform operations against the authentication server. The specified user name must have sufficient permissions to read user and group attributes from the authentication server. This option is only valid with `--type {ldap|ad}` and `--data-access-method {file|object}`. This option along with `--password` is mutually exclusive with `--enable-anonymous-bind`.

In case of `--type ad|ldap` with `--data-access-method object`, the user name must be specified in complete DN format.

--password *userPassword*

Specifies the password of the user name that is specified with the `--user-name` option. This option is only valid with `--type {ldap|ad}` and `--data-access-method {file|object}`.

The password must be in clear text. To hide the password, submit the command without this option and then the system prompts you to enter the password.

--enable-server-tls

Enables TLS communication with the authentication server. This option is disabled by default. For file access configuration, the following certificate file must be placed at: `/var/mmfs/tmp/ldap_cacert.pem` on the current node. For object access configuration, the following certificate file must be placed at: `/var/mmfs/tmp/object_ldap_cacert.pem` on the current node.

If `--data-access-method` is `object`, this option is only valid with `--type {ldap|ad}` and if the `--data-access-method` is `file`, this option is only valid with `--type {ldap}`.

--enable-nfs-kerberos

Enables Kerberized NFSv4-based access to exports. Kerberized NFSv4-based access is only supported for users from AD domains which are configured for fetching UID / GID information from Active Directory (RFC2307 schema attributes). Such an AD domain definition is specified via the `--unixmap-domains` option.

This option is only valid with `--type {ad}` and `--data-access-method {file}`. This option is disabled by default.

--user-dn *ldapUserDN*

Specifies the LDAP group DN. Restricts search of groups within the specified sub-tree. For CIFS access, the value of this parameter is ignored and a search is performed on the baseDN.

This option is only valid with `--type {ldap}` and `--data-access-method {file}`. If this parameter is not set, the system uses the value that is set for baseDN as the default value.

--group-dn *ldapGroupDN*

Specifies the LDAP group suffix. Restricts search of groups within a specified sub-tree.

This option is only valid with `--type {ldap}` and `--data-access-method {file}`. If this parameter is not set, the system uses the value that is set for baseDN as the default value.

--netgroup-dn *ldapGroupDN*

Specifies the LDAP netgroup suffix. The system searches the netgroups based on this suffix. The value must be specified in complete DN format.

This option is only valid with `--type {ldap}` and `--data-access-method {file}`. Default value is baseDN.

--user-objectclass *userObjectClass*

Specifies the object class of user on the authentication server. Only users with specified object class along with other filter are treated as valid users.

If the `--data-access-method` is `object`, this option is only valid with `--type {ldap|ad}`.

If the `--data-access-method` is `file`, this option is only valid with `--type {ldap}`. With `--type ldap`, the default value is `posixAccount` and with `--type ad` the default value is `organizationalPerson`.

--group-objectclass *groupObjectClass*

Specifies the object class of group on the authentication server. This option is only valid with `--type {ldap}` and `--data-access-method {file}`.

--netbios-name *netBiosName*

Specifies the unique identifier of the resources on a network that are running NetBIOS. This option is only valid with `--type {ad|ldap}` and `--data-access-method {file}`.

The NetBIOS name is limited to 15 ASCII characters and must not contain any white space or one of the following characters: / : * ? . " ; |

If AD is selected as the authentication method, the NetBIOS name must be selected carefully. If there are name collisions across multiple IBM Spectrum Scale clusters, or between the AD Domain and the NetBIOS name, the configuration does not work properly. Consider the following points while planning for a naming strategy:

- There must not be NetBIOS name collision between two IBM Spectrum Scale clusters that are configured against the same Active Directory server.
- The domain join of the latter machines revokes the join of the former one.
- The NetBIOS name and the domain name must not collide.
- The NetBIOS name and the short name of the Domain Controllers hosting the domain must not collide.

--domain *domainName*

Specifies the name of the NIS domain. This option is only valid with `--type {nis}` and `--data-access-method {file}`.

The NIS domain that is specified must be served by one of the servers specified with `--server`. This option is mandatory when NIS-based authentication is configured for file access.

--idmap-role *{master|subordinate}*

Specifies the ID map role of the IBM Spectrum Scale system. ID map role of a stand-alone or singular system deployment must be selected "master". The value of the ID map role is important in AFM-based deployments.

This option is only valid with `--type {ad}` and `--data-access-method {file}`.

You can use AD with automatic ID mapping to set up two or more storage subsystems in AFM relationship. The two or more systems configured in a master-subordinate relationship provides a means to synchronize the UIDs and GIDs generated for NAS clients on one system with UIDs and GIDs on the other systems. In the AFM relationship, only one system can be configured as master and other systems must be configured as subordinates. The ID map role of master and subordinate systems are the following:

- **Master:** System creates ID maps on its own.
- **Subordinate:** System does not create ID maps on its own. ID maps must be exported from the master to the subordinate.

While using automatic ID mapping, in order to have same ID maps on systems sharing AFM relationship, you need to export the ID mappings from master to subordinate. The NAS file services are inactive on the subordinate system. If you need to export and import ID maps from one system to another, contact the IBM Support Center.

--idmap-range *lowerValue-higherValue*

Specifies the range of values from which the IBM Spectrum Scale UIDs and GIDs are assigned by the system to the Active Directory users and groups. This option is only valid with `--type {ad}` and `--data-access-method {file}`. The default value is 10000000-299999999. The lower value of the range must be at least 1000. After configuring the IBM Spectrum Scale system with AD authentication, only the higher value can be increased (this essentially increases the number of ranges).

mmuserauth

--idmap-range-size *rangeSize*

Specifies the total number of UIDs and GIDs that are assignable per domain. For example, if `--idmap-range` is defined as 10000000-299999999, and range size is defined as 1000000, 290 domains can be mapped, each consisting of 1000000 IDs.

Choose a value for range size that allows for the highest anticipated RID value among all of the anticipated AD users and AD groups in all of the anticipated AD domains. Choose the range size value carefully because range size cannot be changed after the first AD domain is defined on the IBM Spectrum Scale system.

This option is only valid with `--type {ad}` and `--data-access-method {file}`. Default value is 1000000.

--unixmap-domains *unixDomainMap*

Specifies the AD domains for which user ID and group ID should be fetched from the AD server. This option is only valid with `--type {ad}` and `--data-access-method {file}`. The `unixDomainMap` takes value in this format: `DOMAIN1(L1-H1) [;DOMAIN2(L2-H2) [;DOMAIN3(L3-H3) ...]`

DOMAIN

Use `DOMAIN` to specify an AD domain for which ID mapping services are to be configured. The name of the domain to be specified must be the NetBIOS domain name. The UIDs and GIDs of the users and groups for the specified `DOMAIN` are read from the UNIX attributes that are populated in the RFC2307 schema extension of AD server. Any users or groups, from this domain, with missing UID/GID attributes are denied access. Use the L-H format to specify the ID range. All the users or groups from `DOMAIN` that need access to exports need to have their UIDs or GIDs in the specified range.

The specified range should not intersect with:

- The range specified by using the `--idmap-range` option of the command .
- The range specified for other AD `DOMAIN` for which ID mapping needs to be done from Active Directory (RFC2307 schema attributes) specified in `--unixmap-domains` option.
- The range specified for other AD `DOMAIN` for which ID mapping needs to be done from LDAP server specified in the `--ldapmap-domains` option.

The command reports a failure if you attempt to run the command with such configurations. This is intended to avoid ID collisions among users and groups from different domains.

For example,

```
--unixmap-domains "MYDOMAIN1(20000-50000);MYDOMAIN2(100000-200000)"
```

--ldapmap-domains *ldapDomainMap*

Specifies the AD domains for which user ID and group ID should be fetched from a separate standalone LDAP server. This option is only valid with `--type {ad}` and `--data-access-method {file}`. `ldapDomainMap` takes value of the format as follows,

```
DOMAIN1 (type=stand-alone:ldap_srv=ldapServer:range=Range:usr_dn=userDN:grp_dn=groupDN:[bind_dn=bindDN]:[bind_dn_pwd=bindDNpassword]) [;DOMAIN2(type=stand-alone:ldap_srv=ldapServer:range=Range:usr_dn=userDN:grp_dn=groupDN:[bind_dn=bindDN]:[bind_dn_pwd=bindDNpassword]) [;DOMAIN3(type=stand-alone:ldap_srv=ldapServer:range=Range:usr_dn=userDN:grp_dn=groupDN:[bind_dn=bindDN]:[bind_dn_pwd=bindDNpassword]) ...]
```

DOMAIN

Use `DOMAIN` to specify an AD domain for which ID mapping services are to be configured. The name of the domain to be specified must be the Pre-Win2K domain name. The UID and GID of the users and groups for the specified `DOMAIN` are read from the objects stored on LDAP server in RFC2307 schema attributes. Any users or groups, from this domain, with missing UID/GID attributes are denied access.

type

Defines the type of LDAP server to use.

Supported value: stand-alone.

range

Attribute takes value in the L-H format. Defines the user or group from DOMAIN that needs access to exports need to have their UIDs or GIDs in the specified range. The specified range should not intersect with,

- The range specified using `--idmap-range` option of the command
- The range specified for other AD DOMAIN for which ID mapping needs to be done from Active Directory (RFC2307 schema attributes) specified in `--unixmap-domains` option
- The range specified for other AD DOMAIN for which ID mapping needs to be done from LDAP server specified in `--ldapmap-domains` option

This is intended to avoid ID collisions among users and groups from different domains.

ldap_srv

Defines the name or IP address of the LDAP server to fetch the UID or GID for of a user or group records in RFC2307 schema format. The user and group objects should be in RFC2307 schema format. Specifying only single LDAP server is supported.

user_dn

Defines the bind tree on the LDAP server where user objects shall be found.

grp_dn

Defines the bind tree on the LDAP server where the group objects shall be found.

bind_dn

Optional attribute.

Defines the user DN that should be used for authentication against the LDAP server. If not specified anonymous, bind shall be performed against the LDAP server.

bind_dn_pwd

Optional attribute.

Defines the password of the user DN specified in `bind_dn` to be used for authentication against the LDAP server. Must be specified when `bind_dn` attribute is specified for binding with the LDAP server in the DOMAIN definition.

Password cannot contain these special characters such as semicolon (;) or colon (:).

For example,

```
--ldapmap-domains "MYDOMAIN1(type=stand-alone:range=10000-50000:ldap_srv=myldapserver.mydomain.com
:usr_dn=ou=People,dc=mydomain,dc=com:grp_dn=ou=Groups,dc=mydomain,dc=com
:bind_dn=cn=manager,dc=mydomain,dc=com:bind_dn_pwd=MYPASSWORD);MYDOMAIN2(type=stand-alone
:range=70000-100000:ldap_srv=myldapserver.example.com:usr_dn=ou=People,dc=example,dc=com
:grp_dn=ou=Groups,dc=example,dc=com)"
```

--enable-kerberos

Indicates whether to enable Kerberos in the user authentication. Kerberos is a network authentication protocol for client/server applications that uses symmetric key cryptography. User password in clear text format is never sent over a network to ensure security.

This option is only valid with `--type {ldap}` and `--data-access-method {file}`. This option is disabled by default.

Note: If you need to use Kerberos, ensure that the keytab file is also placed under `/var/mmfs/tmp` directory name as "krb5.keytab"; specifically, on the node where the command is run.

--kerberos-server *kerberosServer*

Specifies the Kerberos server. This option is only valid with `--type {ldap}` and `--data-access-method {file}`.

--kerberos-realm *kerberosRealm*

Indicates the Kerberos server authentication administrative domain. The realm name is usually the all-uppercase version of the domain name. This option is case sensitive.

mmuserauth

--user-name-attrib *UserNameAttribute*

Specifies the attribute to be used to search for user name on authentication server.

If the `--data-access-method` is `object`, this option is only valid with `--type {ldap|ad}`.

If the `--data-access-method` is `file`, this option is only valid with `--type {ldap}`. With `--type ldap`, default value is `cn` and with `--type ad`, the default value is `sAMAccountName`.

--user-id-attrib *UserIDAttribute*

Specifies the attribute to be used to search for user ID on the authentication server.

If `--data-access-method` is `object`, this option is only valid with `--type {ldap|ad}`.

If `--data-access-method` is `file`, this option is only valid with `--type {ldap}`. For `--type ldap`, default value is `uid` and for `--type ad` the default value is `CN`.

--user-mail-attrib *UserMailAttribute*

Specifies the attribute to be used to search for email on authentication server. If the `--data-access-method` is `object`, this option is only valid with `--type {ldap|ad}`. For `--data-access-method file`, this option is only valid with `--type {ldap}`. Default value is `mail`.

--user-filter *userFilter*

Specifies the additional filter to be used to search for user in the authentication server. The filter must be specified in LDAP filter format. This option is only valid with `--type {ldap|ad}` and `--data-access-method {object}`. By default, no filter is used.

--ks-dns-name *keystoneDnsName*

Specifies the DNS name for keystone service. The specified name must be resolved on all protocol nodes for proper functioning. This is optional with `--data-access-method {object}`. If the value is not specified for this parameter, the **mmuserauth service create** command uses the value that is used during the IBM Spectrum Scale system installation.

--ks-admin-user *keystoneAdminName*

Specifies the Keystone server administrative user. This user must be a valid user on authentication server if `--type {ldap|ad}` is specified. In case of `--type local`, new user along with the password specified in `--ks-admin-pwd` is created, and admin role is assigned in Keystone. This option is mandatory with `--data-access-method {object}`.

For `--type {ldap|ad}`, do not specify user name in DN format for `--ks-admin-user`. The name must be the base or short name that is written against the specified `user-id-attrib` or `user-name-attrib` of user on the LDAP server.

--ks-admin-pwd *keystoneAdminPwd*

Specifies the password of the Keystone administrative user. This option is mandatory and valid with `--type {local}` and `--data-access-method {object}`. To hide the password due to security reasons, call the command without this option and the command prompts to enter the password when the **mmuserauth service create** command is issued.

--enable-ks-ssl

Specifies whether to enable SSL for Keystone. Using SSL certificate provides a secured way to access the Keystone service over the HTTPS protocol. This option is only valid with `--data-access-method {object}`. It is disabled by default. If SSL is not enabled for Keystone, the Keystone communicates through HTTP protocol and it results in security risks.

If `--type local | ad | ldap`, keep the valid certificate files at the following location on the current node:

The certificate at `/var/mmfs/tmp/ssl_cert.pem`.

The private key at: `/var/mmfs/tmp/ssl_cert.pem`.

The cacert at: `/var/mmfs/tmp/ssl_cacert.pem`.

If `--type userdefined`, keep the valid certificate files at the following location on the current node:

The cacert at: /var/mmfs/tmp/ssl_cacert.pem.

--ks-swift-user *keystoneSwiftName*

Specifies the username to be used as swift user in proxy-server.conf. If AD or LDAP-based authentication is used, this user must be available in the AD or LDAP authentication server. If local authentication method is used, new user with this name is created in the local database. This option is only valid with `--data-access-method {object}`.

For `--type {ldap|ad}`, do not specify user name in DN format for `--ks-swift-user`. The name must be the base or short name that is written against the specified `user-id-attr` or `user-name-attr` of user on the LDAP server.

--ks-swift-pwd *keystoneSwiftPwd*

Specifies the password of the `ks-swift-user`. If AD or LDAP-based authentication is used, this must be set for `ks-swift-user` in AD or LDAP server. If local authentication method is used, the `ks-swift-user` with this password is created in the local database. This option is only valid with `--data-access-method {object}`.

--enable-ks-casigning

Indicates whether to use a CA signed certificate for PKI (signing). This option is only valid with `--data-access-method {object}` and `--type {ad|ldap|local}`

Valid certificate files must exist at the following location on the current node: /var/mmfs/tmp/signing_cert.pem

Private key at: /var/mmfs/tmp/signing_key.pem

cacert at: /var/mmfs/tmp/signing_cacert.pem

--ks-ext-endpoint *externalEndpoint*

Specifies the endpoint URL of external keystone. Only API v3 and HTTP are supported. This option is only valid with `--data-access-method {object}` and `--type {userdefined}`

--idmapdelete

Specifies whether to delete ID maps. You cannot delete both authentication and ID maps together. The authentication must be deleted first and then ID maps. This option is only valid with **mmuserauth service remove** command.

-N|--nodes *{node-list|cesNodes}*

Verifies the authentication configuration on each node. If the specified node is not protocol node, it is ignored. If protocol node is specified, then the system checks configuration on all protocol nodes. If you do not specify a node, the system checks the configuration of only the current node.

-Y Creates parsable output. This is optional.

-r|--rectify

Rectifies the authentication configurations and missing SSL and TLS certificates.

--server-reachability

Without this flag, the **mmuserauth service check** command only validates whether the authentication configuration files are consistent across the protocol nodes. Use this flag to ensure if the external authentication server is reachable by each protocol node.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **mmuserauth** command.

mmuserauth

The node on which the command is issued must be able to run remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. For more information, see *Requirements for administering a file system* in *IBM Spectrum Scale: Administration and Programming Reference*.

Examples

1. To configure Microsoft Active Directory (AD) based authentication with automatic ID mapping for file access, issue this command:

```
# mmuserauth service create --type ad --data-access-method file --netbios-name
ess --user-name administrator --idmap-role master --servers myADserver
--password Passw0rd --idmap-range-size 1000000 --idmap-range 10000000-299999999
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : AD
PARAMETERS                VALUES
-----
ENABLE_NFS_KERBEROS      false
SERVERS                  myADserver
USER_NAME                 administrator
NETBIOS_NAME             ess
IDMAP_ROLE               master
IDMAP_RANGE              10000000-299999999
IDMAP_RANGE_SIZE         1000000
UNIXMAP_DOMAINS          none

OBJECT access not configured
PARAMETERS                VALUES
-----
```

2. To configure Microsoft Active Directory (AD) based authentication with RFC2307 ID mapping for file access, issue this command:

```
# mmuserauth service create --type ad --data-access-method file
--netbios-name ess --user-name administrator --idmap-role master
--servers myAdserver --password Passw0rd --idmap-range-size 1000000
--idmap-range 10000000-299999999 --unixmap-domains 'DOMAIN(5000-20000)'
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : AD
PARAMETERS                VALUES
-----
ENABLE_NFS_KERBEROS      false
SERVERS                  myAdserver
USER_NAME                 administrator
NETBIOS_NAME             ess
IDMAP_ROLE               master
IDMAP_RANGE              10000000-299999999
```



```
IDMAP_RANGE_SIZE      1000000
UNIXMAP_DOMAINS      DOMAIN(5000-20000)
```

```
OBJECT access not configured
PARAMETERS          VALUES
```

- To configure Microsoft Active Directory (AD) based authentication with LDAP ID mapping for file access, issue this command:

```
mmuserauth service create --data-access-method file --type ad --servers myADserver
--user-name administrator --password Passw0rd --netbios-name ess --idmap-role master
--ldapmap-domains "SONAS(type=stand-alone: range=1000-10000:ldap_srv=9.118.46.17:
usr_dn=ou=People,dc=example,dc=com:grp_dn=ou=Groups,dc=example,dc=com:bind_dn=cn=manager,
dc=example,dc=com:bind_dn_pwd=password)"
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : AD
PARAMETERS          VALUES
-----
ENABLE_NFS_KERBEROS  false
SERVERS              myADserver
USER_NAME            administrator
NETBIOS_NAME         ess
IDMAP_ROLE           master
IDMAP_RANGE          10000000-299999999
IDMAP_RANGE_SIZE     1000000
UNIXMAP_DOMAINS      none
LDAPMAP_DOMAINS      DOMAIN(type=stand-alone: range=1000-10000:
ldap_srv=myLDAPserver:usr_dn=ou=People,dc=example,dc=com:grp_dn=
ou=Groups,dc=example,dc=com:bind_dn=cn=manager,dc=example,dc=com)
```

```
OBJECT access not configured
PARAMETERS          VALUES
-----
```

- To configure Microsoft Active Directory (AD) based authentication with LDAP ID mapping for file access (anonymous binding with LDAP), issue this command:

```
# mmuserauth service create --data-access-method file --type ad
--servers myADserver --user-name administrator --password Passw0rd
--netbios-name ess --idmap-role master --ldapmap-domains
"SONAS(type=stand-alone: range=1000-10000:ldap_srv=9.118.46.17:
usr_dn=ou=People,dc=example,dc=com:grp_dn=ou=Groups,dc=example,dc=com)"
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : AD
PARAMETERS          VALUES
-----
ENABLE_NFS_KERBEROS  false
SERVERS              myADserver
```

mmuserauth

```
USER_NAME      administrator
NETBIOS_NAME   ess
IDMAP_ROLE     master
IDMAP_RANGE    10000000-299999999
IDMAP_RANGE_SIZE 1000000
UNIXMAP_DOMAINS none
LDAPMAP_DOMAINS DOMAIN(type=stand-alone: range=1000-10000:ldap_srv=myLDAPserver:
usr_dn=ou=People,dc=example,dc=com:grp_dn=ou=Groups,dc=example,dc=com)
```

```
OBJECT access not configured
PARAMETERS      VALUES
-----
```

5. To configure LDAP-based authentication with TLS encryption for file access, issue this command:

```
# mmuserauth service create --type ldap --data-access-method file
--servers es-pune-host-01 --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-server-tls
```

The system displays output similar to this:

File Authentication configuration completed successfully.

Note: Before issuing the **mmuserauth service create** command to configure LDAP with TLS, ensure that the CA certificate for LDAP server is placed under `/var/mmfs/tmp` directory with the name "ldap_cacert.pem" specifically on the protocol node where the command is issued.

To verify the authentication configuration, use the **mmuserauth service list** as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : LDAP
PARAMETERS      VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS      true
ENABLE_KERBEROS        false
USER_NAME             cn=manager,dc=example,dc=com
SERVERS               myLDAPserver
NETBIOS_NAME          ess
BASE_DN               dc=example,dc=com
USER_DN               none
GROUP_DN              none
NETGROUP_DN           none
USER_OBJECTCLASS      posixAccount
GROUP_OBJECTCLASS     posixGroup
USER_NAME_ATTRIB      cn
USER_ID_ATTRIB        uid
KERBEROS_SERVER       none
KERBEROS_REALM        none
```

```
OBJECT access not configured
PARAMETERS      VALUES
-----
```

6. To configure LDAP-based authentication with Kerberos for file access, issue this command:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-kerberos
--kerberos-server myKerberosServer --kerberos-realm example.com
```

The system displays output similar to this:

File Authentication configuration completed successfully.

Note: Before issuing the **mmuserauth service create** command to configure LDAP with Kerberos, ensure that the keytab file is also placed under `/var/mmfs/tmp` directory name as "krb5.keytab" specifically on the node where the command is run.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : LDAP
PARAMETERS                VALUES
-----
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        false
ENABLE_KERBEROS           true
USER_NAME                 cn=manager,dc=example,dc=com
SERVERS                   myLDAPserver
NETBIOS_NAME              ess
BASE_DN                   dc=example,dc=com
USER_DN                   none
GROUP_DN                  none
NETGROUP_DN              none
USER_OBJECTCLASS          posixAccount
GROUP_OBJECTCLASS         posixGroup
USER_NAME_ATTRIB         cn
USER_ID_ATTRIB            uid
KERBEROS_SERVER           myKerberosServer
KERBEROS_REALM            example.com
```

OBJECT access not configured

```
PARAMETERS                VALUES
-----
```

- To configure LDAP with TLS and Kerberos for file access, issue this command:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-server-tls --enable-kerberos
--kerberos-server myKerberosServer --kerberos-realm example.com
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : LDAP
PARAMETERS                VALUES
-----
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        true
ENABLE_KERBEROS           true
USER_NAME                 cn=manager,dc=example,dc=com
SERVERS                   es-pune-host-01
NETBIOS_NAME              ess
BASE_DN                   dc=example,dc=com
USER_DN                   none
GROUP_DN                  none
NETGROUP_DN              none
USER_OBJECTCLASS          posixAccount
GROUP_OBJECTCLASS         posixGroup
USER_NAME_ATTRIB         cn
```

mmuserauth

```
USER_ID_ATTRIB      uid
KERBEROS_SERVER    myKerberosServer
KERBEROS_REALM     example.com
```

```
OBJECT access not configured
PARAMETERS          VALUES
-----
```

8. To configure LDAP without TLS and Kerberos for file access, issue this command:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com --user-name
cn=manager,dc=example,dc=com --password secret --netbios-name ess
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : LDAP
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND  false
ENABLE_SERVER_TLS      false
ENABLE_KERBEROS        false
USER_NAME              cn=manager,dc=example,dc=com
SERVERS                myLDAPserver
NETBIOS_NAME           ess
BASE_DN                dc=example,dc=com
USER_DN                none
GROUP_DN               none
NETGROUP_DN           none
USER_OBJECTCLASS       posixAccount
GROUP_OBJECTCLASS      posixGroup
USER_NAME_ATTRIB      cn
USER_ID_ATTRIB        uid
KERBEROS_SERVER        none
KERBEROS_REALM        none
```

```
OBJECT access not configured
PARAMETERS          VALUES
-----
```

9. To configure NIS-based authentication for file access, issue this command:

```
# mmuserauth service create --type nis --data-access-method file
--servers myNISserver --domain nisdomain
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : NIS
PARAMETERS          VALUES
-----
SERVERS              myNISserver
DOMAIN               nisdomain
```

```
OBJECT access not configured
PARAMETERS          VALUES
-----
```

10. To configure user-defined authentication for file access, issue this command:

```
# mmuserauth service create --data-access-method file --type userdefined
```

The system displays output similar to this:

File Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access configuration : USERDEFINED
PARAMETERS          VALUES
-----
```

```
OBJECT access not configured
PARAMETERS          VALUES
-----
```

11. To configure local authentication for object access, issue this command:

```
# mmuserauth service create --data-access-method object --type local
--ks-dns-name ksDNSname --ks-admin-user admin
--ks-admin-pwd Passw0rd
```

The system displays output similar to this:

Object configuration with local (Database) as identity backend is completed successfully.

Object Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access not configured
PARAMETERS          VALUES
-----
```

```
OBJECT access configuration : LOCAL
PARAMETERS          VALUES
-----
```

```
ENABLE_KS_SSL          false
ENABLE_KS_CASIGNING    false
KS_ADMIN_USER          admin
```

12. To configure AD without TLS authentication for object access, issue this command:

```
# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=example,dc=com" --password Passw0rd --base-dn "dc=example,DC=com"
--ks-dns-name ksDNSname --ks-admin-user admin --servers myADserver --user-id-attr cn
--user-name-attr sAMAccountName --user-objectclass organizationalPerson --user-dn "cn=Users,dc=example,dc=com"
--ks-swift-user swift --ks-swift-pwd Passw0rd
```

The system displays output similar to this:

Object configuration with LDAP (Active Directory) as identity backend is completed successfully.

Object Authentication configuration completed successfully.

mmuserauth

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration: AD
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS       false
ENABLE_KS_SSL           false
USER_NAME               cn=Administrator,cn=Users,dc=example,dc=com
SERVERS                 myADserver
BASE_DN                  dc=IBM,DC=local
USER_DN                  cn=users,dc=example,dc=com
USER_OBJECTCLASS         organizationalPerson
USER_NAME_ATTRIB        sAMAccountName
USER_ID_ATTRIB          cn
USER_MAIL_ATTRIB        mail
USER_FILTER             none
ENABLE_KS_CASIGNING     false
KS_ADMIN_USER           admin
```

13. To configure AD with TLS authentication for object access, issue this command:

```
# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=example,dc=com" --password Passw0rd --base-dn
"dc=example,DC=com" --enable-server-tls --ks-dns-name ksDNSname --ks-admin-user admin --servers
myADserver --user-id-attrib cn --user-name-attrib sAMAccountName --user-objectclass organizationalPerson
--user-dn "cn=Users,dc=example,dc=com" --ks-swift-user swift --ks-swift-pwd Passw0rd
```

The system displays output similar to this:

```
Object configuration with LDAP (Active Directory) as identity backend is completed
successfully.
Object Authentication configuration completed successfully.
```

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```
FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration: AD
PARAMETERS          VALUES
-----
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS       true
ENABLE_KS_SSL           false
USER_NAME               cn=Administrator,cn=Users,dc=example,dc=com
SERVERS                 myADserver
BASE_DN                  dc=IBM,DC=local
USER_DN                  cn=users,dc=example,dc=com
USER_OBJECTCLASS         organizationalPerson
USER_NAME_ATTRIB        sAMAccountName
USER_ID_ATTRIB          cn
```

```

USER_MAIL_ATTRIB      mail
USER_FILTER           none
ENABLE_KS_CASIGNING   false
KS_ADMIN_USER         admin

```

14. To configure LDAP-based authentication for object access, issue this command:

```

# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=example,dc=com" --password "Passw0rd"
--base-dn dc=example,dc=com --ks-dns-name ksDNSname --ks-admin-user admin --servers myLDAPserver
--user-dn "ou=People,dc=example,dc=com"
--ks-swift-user swift --ks-swift-pwd Passw0rd

```

The system displays output similar to this:

Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```

FILE access not configured
PARAMETERS          VALUES
-----
OBJECT access configuration : LDAP
PARAMETERS          VALUES
-----
ENABLE_SERVER_TLS   false
ENABLE_KS_SSL       false
USER_NAME           cn=manager,dc=example,dc=com
SERVERS             myLDAPserver
BASE_DN             dc=example,dc=com
USER_DN             ou=people,dc=example,dc=com
USER_OBJECTCLASS    posixAccount
USER_NAME_ATTRIB   cn
USER_ID_ATTRIB      uid
USER_MAIL_ATTRIB    mail
USER_FILTER         none
ENABLE_KS_CASIGNING false
KS_ADMIN_USER       admin

```

15. To configure LDAP with TLS-based authentication for object access, issue this command:

```

# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=example,dc=com" --password "Passw0rd"
--base-dn dc=example,dc=com --enable-server-tls
--ks-dns-name ksDNSname --ks-admin-user admin --servers myLDAPserver
--user-dn "ou=People,dc=example,dc=com" --ks-swift-user swift
--ks-swift-pwd Passw0rd

```

The system displays output similar to this:

Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays output similar to this:

```

FILE access not configured
PARAMETERS          VALUES
-----

```

mmuserauth

```
OBJECT access configuration : LDAP
PARAMETERS                   VALUES
-----
ENABLE_SERVER_TLS            true
ENABLE_KS_SSL                 false
USER_NAME                     cn=manager,dc=example,dc=com
SERVERS                       myLDAPserver
BASE_DN                       dc=example,dc=com
USER_DN                       ou=people,dc=example,dc=com
USER_OBJECTCLASS              posixAccount
USER_NAME_ATTRIB              cn
USER_ID_ATTRIB                uid
USER_MAIL_ATTRIB              mail
USER_FILTER                   none
ENABLE_KS_CASIGNING           false
KS_ADMIN_USER                 admin
```

16. To remove the authentication method that is configured for file access, issue this command:

```
# mmuserauth service remove --data-access-method file
```

The system displays output similar to this:

```
mmuserauth service remove: Command successfully completed
```

Note: Authentication configuration and ID maps cannot be deleted together. To remove ID maps, remove the authentication configuration first and then remove ID maps. Also, you cannot delete ID maps that are used for file and object access together. That is, when you delete the ID maps, the value that is specified for `--data-access-method` must be either `file` or `object`.

17. To remove the authentication method that is configured for object access, issue this command:

```
# mmuserauth service remove --data-access-method object
```

The system displays output similar to this:

```
mmuserauth service remove: Command successfully completed
```

Note: Authentication configuration and ID maps cannot be deleted together. To remove ID maps, remove the authentication configuration first and then remove the ID maps. Also, you cannot delete ID maps that are used for file and object access together. That is, when you delete the ID maps, the value that is specified for `--data-access-method` must be either `file` or `object`.

18. To check whether the authentication configuration is consistent across the cluster and the required services are enabled and running, issue this command:

```
# mmuserauth service check --data-access-method file --nodes cesNodes --rectify
```

The system displays output similar to this:

```
Userauth file check on node: dgnode3
  Checking SSSD_CONF: OK
  LDAP server status
  LDAP server 192.0.2.18 : OK
Service 'sssd' status: OK
Userauth file check on node: dgnode2
dgnode2: not CES node. Ignoring...
```

19. To check whether the file authentication configuration is consistent across the cluster and the required services are enabled and running, and if you do not want to correct the situation, issue this command:

```
# mmuserauth service check --data-access-method file --nodes cesNodes --rectify
```

20. To check that all object configuration files (including certificates) are present, and if not, rectify the situation by issuing the following command:

```
# mmuserauth service check --data-access-method object rectify
```

The system displays output similar to this:


```

Userauth object check on node: node1
Checking keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK
Checking /etc/keystone/ssl/certs/object_ldap_cacert.pem: OK
Service 'openstack-keystone' status: OK

```

21. To check if the external authentication server is reachable by each protocol node, issue the following command:

```
mmuserauth service check --server-reachability
```

- a. If file is not configured, object is configured, and there are no errors, the system displays output similar to this:

```

Userauth object check on node: vmnode2
Checking keystone.conf: OK
Checking wsgi-keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK

```

```
LDAP servers status
```

```
LDAP server 9.118.37.234 : OK
Service 'httpd' status: OK
```

- b. If file is not configured, object is configured, and there is a single error, the system displays output similar to this:

```

Userauth object check on node: vmnode2
Checking keystone.conf: OK
Checking wsgi-keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK

```

```
LDAP servers status
```

```
LDAP server 9.118.37.234 : ERROR
Service 'httpd' status: OK
```

- c. If file and object are configured and there are no errors, the system displays output similar to this:

```

Userauth file check on node: vmnode2
Checking nsswitch file: OK

```

```
AD servers status
```

```
NETLOGON connection: OK
Domain join status: OK
Machine password status: OK
Service 'gpfs-winbind' status: OK
```

```
Userauth object check on node: vmnode2
```

```

Checking keystone.conf: OK
Checking wsgi-keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK

```

```
LDAP servers status
```

```
LDAP server 9.118.37.234 : OK
Service 'httpd' status: OK
```

- d. If file and object are configured and there is a single error, the system displays output similar to this:

```

Userauth file check on node: vmnode2
Checking nsswitch file: OK

```

```
AD servers status
```

```
NETLOGON connection: OK
```

mmuserauth

```
Domain join status: OK
Machine password status: ERROR
Service 'gpfs-winbind' status: OK
```

```
Userauth object check on node: vmnode2
```

```
Checking keystone.conf: OK
Checking wsgi-keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK
```

```
LDAP servers status
LDAP server 9.118.37.234 : OK
```

- e. If file and object are configured and there are multiple errors, the system displays output similar to this:

```
Userauth file check on node: vmnode2
Checking nsswitch file: OK
```

```
AD servers status
NETLOGON connection: ERROR
Domain join status: ERROR
Machine password status: ERROR
Service 'gpfs-winbind' status: OK
```

```
Userauth object check on node: vmnode2
Checking keystone.conf: OK
Checking wsgi-keystone.conf: OK
Checking /etc/keystone/ssl/certs/signing_cert.pem: OK
Checking /etc/keystone/ssl/private/signing_key.pem: OK
Checking /etc/keystone/ssl/certs/signing_cacert.pem: OK
```

```
LDAP servers status
LDAP server 9.118.37.234 : ERROR
Service 'httpd' status: OK
```

Note: The `--rectify` or `-r` option cannot fix server reachability errors. Specifying that option with `--server-reachability` may fix the erroneous config files and service-related errors only.

See also

- “mmces command” on page 304
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmobj command” on page 581
- “mmsmb command” on page 663

Location

```
/usr/lpp/mmfs/bin
```

mmwinservctl command

Manages the **mmwinserv** Windows service.

Synopsis

```
mmwinservctl set [--account AccountName [--password Password]] [--remote-shell {yes | no}]
[-N {Node[,Node...]} | NodeFile | NodeClass] [-v]
```

or

```
mmwinservctl {enable | disable | query} [-N {Node[,Node...]} | NodeFile | NodeClass] [-v]
```

Availability

Available on all IBM Spectrum Scale editions. Available on Windows.

Description

mmwinserv is a GPFS for Windows service that is needed for the proper functioning of the GPFS daemon on nodes running Windows. Optionally, the service can be configured to provide a remote execution facility for GPFS administration commands.

Use the **mmwinservctl** command to manage the **mmwinserv** service. You can set the log on account and password for the service, enable or disable the service, enable or disable the service's remote execution facility, or query its current state.

The **mmwinservctl** command must be run on a Windows node and it has no effect on nodes running other operating systems.

If the remote execution facility of **mmwinserv** is enabled, a Windows GPFS cluster can be configured to use **mmwinrsh** and **mmwinrcp** as the remote shell and remote file copy commands:

- **mmwinrsh** (`/usr/lpp/mmfs/bin/mmwinrsh`) uses Windows Named Pipes to pass the command to the target node.
- **mmwinrcp** (`/usr/lpp/mmfs/bin/mmwinrcp`) is a wrapper module that invokes the Cygwin **cp** command to copy the files that are needed by the **mm** commands. The path names on remote hosts are translated into path names based on the standard Windows ADMIN\$ share.

An account must be given the right to log on as a service before it can be used to run **mmwinserv**. The right to log on as a service is controlled by the Local Security Policy of each Windows node. You can use the Domain Group Policy to set the Local Security Policy on all Windows nodes in a GPFS cluster.

For more information on the **mmwinserv** service, see the topic “Configuring the GPFS Administration service” in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Parameters

set

Sets the service configuration options and restarts the service if it is running.

enable

Sets the service to automatic startup and starts the service.

disable

Sets the service to disabled and stops the service.

query

Returns information about the service's configuration and current state.

mmwinservctl

--account *AccountName*

Specifies the log on account name for the **mmwinserv** service. By default, **mmwinserv** is configured to run using the **LocalSystem** account.

--password *Password*

Specifies the log on password for the **mmwinserv** service.

--remote-shell {**yes** | **no**}

Specifies whether or not remote connections are allowed.

-N {*Node[,Node...]* | *NodeFile* | *NodeClass*}

Specifies the list of nodes on which to perform the action. The default is the node on which the **mmwinservctl** command is issued.

If the node on which the **mmwinservctl** command is issued belongs to a GPFS cluster, the nodes specified with the **-N** parameter must belong to the cluster.

If the node on which the **mmwinservctl** command is issued does not belong to a GPFS cluster, the nodes specified with the **-N** parameter must be identified by their host names or IP addresses. Node classes and node numbers cannot be used.

For general information on how to specify node names, see *Specifying nodes as input to GPFS commands* in the *IBM Spectrum Scale: Administration and Programming Reference*.

-v Displays progress and intermediate error messages.

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must be a member of the Domain Admins group to run the **mmwinservctl** command.

Examples

1. To specify 'gpfs\root' as the log on account name for the **mmwinserv** service and enable the remote command execution facility on nodes **ls21n19** and **ls21n20**, issue:

```
mmwinservctl set -N ls21n19,ls21n20 --account gpfs/root -password abcdefg --remote-shell yes
```

The system displays information similar to:

Node name	Service state	Remote shell	Account name
ls21n19	START_PENDING	yes	gpfs\root
ls21n20	START_PENDING	yes	gpfs\root

2. To display the current state of the **mmwinserv** service on all nodes in the cluster, issue:

```
mmwinservctl query -N all
```

The system displays information similar to:

Node name	Service state	Remote shell	Account name
ls21n19	RUNNING	yes	gpfs\root
ls21n20	RUNNING	yes	gpfs\root
ls21n14	RUNNING	yes	LocalSystem

Location

/usr/lpp/mmfs/bin

spectrumscale command

Installs and configures GPFS; adds nodes to a cluster; deploys and configures protocols, performance monitoring tools, and authentication services; and upgrades GPFS and protocols.

Synopsis

```
spectrumscale setup [-i SSHIdentity] [-s ServerIP] [--storesecret]
```

or

```
spectrumscale node add [-g] [-q] [-m] [-a] [-n] [-p [ExportIP]] Node
```

or

```
spectrumscale node load [-q] [-m] [-a] [-n] NodeFile
```

or

```
spectrumscale node delete [-f] Node
```

or

```
spectrumscale node clear [-f]
```

or

```
spectrumscale node list
```

or

```
spectrumscale config gpfs [-c ClusterName] [-p {default | randomio}]
    [-r RemoteShell] [-rc RemoteFileCopy]
    [-e EphemeralPortRange]
```

or

```
spectrumscale config protocols [-l] [-f FileSystem] [-m MountPoint] [-e ExportIPPool]
```

or

```
spectrumscale config object [-f FileSystem] [-m MountPoint] [-e EndPoint] [-o ObjectBase]
    [-i InodeAllocation] [-t AdminToken]
    [-au AdminUser] [-ap AdminPassword]
    [-su SwiftUser] [-sp SwiftPassword]
    [-dp DatabasePassword]
    [-mr MultiRegion] [-rn RegionNumber]
    [-s3 {on | off}]
```

or

```
spectrumscale config ntp [-e {on | off}] [-l List] [-s Upstream_Servers]
```

or

```
spectrumscale nsd add -p Primary [-s Secondary] [-fs FileSystem]
    [-po Pool]
    [-u {dataOnly | dataAndMetadata | metaDataOnly | descOnly | localCache}]
    [-fg FailureGroup] [--no-check]
    PrimaryDevice [PrimaryDevice ...]
```

or

```
spectrumscale nsd balance [--node Node | --all]
```

spectrumscale

or

```
spectrumscale nsd delete NSD
```

or

```
spectrumscale nsd modify [-n Name]  
                        [-u {dataOnly | dataAndMetadata | metadataOnly | descOnly}]  
                        [-po Pool] [-fs FileSystem] [-fg FailureGroup]  
                        NSD
```

or

```
spectrumscale nsd clear [-f]
```

or

```
spectrumscale nsd list
```

or

```
spectrumscale filesystem modify [-b {64K | 128K | 256K | 512K | 1M | 2M | 4M | 8M | 16M}]  
                                [-m MountPoint]  
                                FileSystem
```

or

```
spectrumscale filesystem list
```

or

```
spectrumscale auth file {ldap | ad | nis | none}
```

or

```
spectrumscale auth object [--https] [--pki] {local | external | ldap | ad}
```

or

```
spectrumscale enable {object | nfs | smb}
```

or

```
spectrumscale disable {object | nfs | smb}
```

CAUTION:

Disabling object service discards OpenStack Swift configuration and ring files from the CES cluster. If Openstack Keystone configuration is configured locally, disabling object storage also discards the Keystone configuration and database files from the CES cluster. However, the data is not removed. For subsequent object service enablement with a clean configuration and new data, remove object store fileset and set up object environment. See the mmobj swift base command. For more information, contact the IBM® Support Center.

or

```
spectrumscale install [-pr] [-po]
```

or

```
spectrumscale deploy [-pr] [-po] [-s]
```

or

```
spectrumscale upgrade [-pr | -po | -ve] [-f]
```

Availability

The **spectrumscale** is available as follows:

- Available with IBM Spectrum Scale Standard Edition or higher.
- Available on Red Hat Enterprise Linux 7 only.

Description

Use the **spectrumscale** command (also called the **spectrumscale** installation toolkit) to do the following:

- Install and configure GPFS.
- Add GPFS nodes to an existing cluster.
- Deploy and configure SMB, NFS, OpenStack Swift, and performance monitoring tools on top of GPFS.
- Configure authentication services for protocols.
- Upgrade GPFS and protocols.

Note: The following prerequisites and assumptions apply:

- The **spectrumscale** installation toolkit requires the following packages:
 - python-2.6 with argparse
 - python-2.7
- TCP traffic from the nodes should be allowed through the firewall to communicate with the install toolkit on port 8889 for communication with the chef zero server and port 10080 for package distribution.
- The nodes themselves have external Internet access or local repository replicas that can be reached by the nodes to install necessary packages (dependency installation). Review the Repository Setup section in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* for more information on these repositories.
- To install protocols, there must a GPFS cluster running a minimum version of 4.1.1.0 with CCR enabled.
- The node that you plan to run the install toolkit from must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages.

Parameters

setup

Installs Chef and its components, as well as configure the install node in the cluster definition file. The IP address passed in should be the node from which the **spectrumscale** installation toolkit will be run. The SSH key passed in should be the key the installer should use to have passwordless SSH onto all other nodes. This is the first command you will run to set up IBM Spectrum Scale. This option accepts the following arguments:

- i** *SSHIdentity*
Adds the path to the SSH identity file into the configuration.
- s** *ServerIP*
Adds the control node IP into the configuration.
- storesecret**
Disables the prompts for the encryption secret.

CAUTION:

If you use this option, passwords will not be securely stored.

This is the first command to run to set up IBM Spectrum Scale.

spectrumscale

node

Used to add, remove, or list nodes in the cluster definition file. This command only interacts with this configuration file and does not directly configure nodes in the cluster itself. The nodes that have an entry in the cluster definition file will be used during install, deploy, or upgrade. This option accepts the following arguments:

add *Node*

Adds the specified node and configures it according to the following arguments:

- g Adds GPFS Graphical User Interface servers to the cluster definition file.
- q Configures the node as a quorum node.
- m Configures the node as a manager node.
- a Configures the node as an admin node.
- n Specifies the node as NSD.
- p [*ExportIP*]
Configures the node as a protocol node and optionally assigns it an IP.

Node

Specifies the node name.

load *NodeFile*

Loads the specified file containing a list of nodes, separated per line; adds the nodes in the file and configures them according to the following:

- q Configures the node as a quorum node.
- m Configures the node as a manager node.
- a Configures the node as an admin node.
- n Specifies the node to be NSD.

delete *Node*

Removes the specified node from the configuration. The following option is accepted.

- f Forces the action without manual confirmation.

clear

Clears the current node configuration. The following option is accepted:

- f Forces the action without manual confirmation.

list

Lists the nodes configured in your environment.

config

Used to set properties in the cluster definition file that will be used during install, deploy, or upgrade. This command only interacts with this configuration file and does not directly configure these properties on the GPFS cluster. This option accepts the following arguments:

gpfs

Sets any of the following GPFS-specific properties to be used during GPFS installation and configuration:

- c *ClusterName*

-p

Specifies the profile to be set on cluster creation. The following values are accepted:

default

Specifies that the **GpfsProtocolDefaults** profile is to be used.

randomio

Specifies that the **GpfsProtocolRandomIO** profile is to be used.

-r RemoteShell

Specifies the remote shell binary to be used by GPFS. If no remote shell is specified in the cluster definition file, `/usr/bin/ssh` will be used as the default.

-rc RemoteFileCopy

Specifies the remote file copy binary to be used by GPFS. If no remote file copy binary is specified in the cluster definition file, `/usr/bin/scp` will be used as the default.

-e EphemeralPortRange

Specifies an ephemeral port range to be set on all GPFS nodes. If no port range is specified in the cluster definition, 60000-61000 will be used as default.

For information about ephemeral port range, see the topic about GPFS port usage in the “Miscellaneous advanced administration tasks” chapter of *IBM Spectrum Scale: Advanced Administration Guide*.

protocols

Provides details of the GPFS environment that will be used during protocol deployment, according to the following options:

-l**-f FileSystem**

Specifies the file system.

-m MountPoint

Specifies the mount point.

-e ExportIPPool

Exports IP pool

object

Sets any of the following Object-specific properties to be used during Object deployment and configuration:

-f FileSystem

Specifies the file system.

-m MountPoint

Specifies the mount point.

-e EndPoint

Specifies the host name that will be used for access to the object store. This should be a round-robin DNS entry that maps to all CES IP addresses or the address of a load balancer front end; this will distribute the load of all keystone and object traffic that is routed to this host name. Therefore the endpoint is an IP address in a DNS or in a load balancer that maps to a group of export IPs (that is, CES IPs that were assigned on the protocol nodes).

-o ObjectBase

Specifies the object base.

-i InodeAllocation

Specifies the inode allocation.

-t AdminToken

Specifies the admin token.

-au AdminUser

Specifies the user name for the admin.

spectrumscale

-ap *AdminPassword*
Specifies the admin user password.

Note: You will be prompted to enter a Secret Encryption Key which will be used to securely store the password. Choose a memorable pass phrase which you will be prompted for each time you enter the password.

-su *SwiftUser*
Specifies the Swift user name.

-sp *SwiftPassword*
Specifies the Swift user password.

Note: You will be prompted to enter a Secret Encryption Key which will be used to securely store the password. Choose a memorable pass phrase which you will be prompted for each time you enter the password.

-dp *DataBasePassword*
Specifies the object database.

Note: You will be prompted to enter a Secret Encryption Key which will be used to securely store the password. Choose a memorable pass phrase which you will be prompted for each time you enter the password.

| **-mr** *MultiRegion*
| Enables the **multi-region** option.

| **-rn** *RegionNumber*
| Specifies the region number.

-s3 on | off
Specifies whether s3 is to be turned on or off.

ntp

Used to add, list, or remove NTP nodes to the configuration. NTP nodes will be configured on the cluster as follows: the admin node will point to the upstream NTP servers that you provide to determine the correct time. The rest of the nodes in the cluster will point to the admin node to obtain the time.

-s *Upstream_Server*
Specifies the host name that will be used. You can use an upstream server that you have already configured, but it cannot be part of your Spectrum Scale cluster.

Note: NTP works best with at least four upstream servers. If you provide fewer than four, you will receive a warning during installation advising that you add more.

-l *List*
Allows for viewing your NTP setup.

-e on | off
Specifies whether NTP is enabled or not. If this option is turned to off, you will receive a warning during installation.

nsd

Used to add, remove, list or balance NSDs, as well as add file systems in the cluster definition file. This command only interacts with this configuration file and does not directly configure NSDs on the cluster itself. The NSDs that have an entry in the cluster definition file will be used during install. This option accepts the following arguments:

add

Adds an NSD to the configuration, according to the following specifications:

- p** *Primary*
Specifies the primary NSD server name.
- s** *Secondary*
Specifies the secondary NSD server name. This option can be repeated to specify multiple secondary NSD servers.
- fs** *FileSystem*
Specifies the file system to which the NSD is assigned.
- po** *Pool*
Specifies the file system pool.
- u**
Specifies NSD usage. The following values are accepted:
 - dataOnly**
 - dataAndMetadata**
 - metaDataOnly**
 - descOnly**
 - localCache**
- fg** *FailureGroup*
Specifies the failure group to which the NSD belongs.
- no-check**
Specifies not to check for the device on the server.
- PrimaryDevice*
Specifies the device name on the primary NSD server.

balance

Balances the NSD preferred node between the primary and secondary nodes. The following options are accepted:

- node** *Node*
Specifies the node to move NSDs from when balancing.
- all**
Specifies that all NSDs are to be balanced.

delete *NSD*

Removes the specified NSD from the configuration.

modify *NSD*

Modifies the NSD parameters on the specified NSD, according to the following options:

- n** *Name*
Specifies the name.
- u** The following values are accepted:
 - dataOnly**
 - dataAndMetadata**
 - metadataOnly**
 - descOnly**
- po** *Pool*
Specifies the pool

spectrumscale

-fs *FileSystem*
Specifies the file system.

-fg *FailureGroup*
Specifies the failure group.

clear

Clears the current NSD configuration. The following option is accepted:

-f Forces the action without manual confirmation.

nsd list

Lists the NSDs configured in your environment.

filesystem

Used to list or modify file systems in the cluster definition file. This command only interacts with this configuration file and does not directly modify file systems on the cluster itself. To modify the properties of a file system in the cluster definition file, the file system must first be added with **spectrumscale nsd**. This option accepts the following arguments:

modify

Modifies the file system attributes. This option accepts the following arguments:

-b Specifies the file system block size. This argument accepts the following values: 64K, 128K, 256K, 512K, 1M, 2M, 4M, 8M, 16M.

-m *MountPoint*
Specifies the mount point.

FileSystem
Specifies the file system to be modified.

list

Lists the file systems configured in your environment.

auth

Used to configure either Object or File authentication on protocols in the cluster definition file. This command only interacts with this configuration file and does not directly configure authentication on the protocols. To configure authentication on the GPFS cluster during a deploy, authentication settings must be provided through the use of a template file. This option accepts the following arguments:

file

Specifies file authentication.

One of the following must be specified:

ldap

ad

nis

none

object

Specifies object authentication.

Either of the following options are accepted:

--https

--pki

One of the following must be specified:

local

external

ldap**ad**

Both file and object authentication can be set up with the authentication backend server specified. Running this command will open a template settings file to be filled out before installation.

enable

Used to enable Object, SMB or NFS in the cluster definition file. This command only interacts with this configuration file and does not directly enable any protocols on the GPFS cluster itself. The default configuration is that all protocols are disabled. If a protocol is enabled in the cluster definition file, this protocol will be enabled on the GPFS cluster during deploy. This option accepts the following arguments:

obj

Object

nfs

NFS

smb

SMB

disable

Used to disable Object, SMB or NFS in the cluster definition file. This command only interacts with this configuration file and does not directly disable any protocols on the GPFS cluster itself. The default configuration is that all protocols are disabled, so this command is only necessary if a protocol has previously been enabled in the cluster definition file, but is no longer required.

Note: Disabling a protocol in the cluster definition will not disable this protocol on the GPFS cluster during a deploy, it merely means that this protocol will not be enabled during a deploy.

This option accepts the following arguments:

obj

Object

CAUTION:

Disabling object service discards OpenStack Swift configuration and ring files from the CES cluster. If Openstack Keystone configuration is configured locally, disabling object storage also discards the Keystone configuration and database files from the CES cluster. However, the data is not removed. For subsequent object service enablement with a clean configuration and new data, remove object store fileset and set up object environment. See the `mmobj swift base` command. For more information, contact the IBM® Support Center.

nfs

NFS

smb

SMB

install

Installs, creates a GPFS cluster, creates NSDs and adds nodes to an existing GPFS cluster. The **spectrumscale** installation toolkit will use the environment details in the cluster definition file to perform these tasks. If all configuration steps have been completed, this option can be run with no arguments (and pre-install and post-install checks will be performed automatically).

For a “dry-run,” the following arguments are accepted:

-pr

Performs a pre-install environment check.

spectrumscale

-po

Performs a post-install environment check.

deploy

Creates file systems, deploys protocols, and configures protocol authentication on an existing GPFS cluster. The **spectrumscale** installation toolkit will use the environment details in the cluster definition file to perform these tasks. If all configuration steps have been completed, this option can be run with no arguments (and pre-deploy and post-deploy checks will be performed automatically). However, the secret key will be prompted for unless it is passed in as an argument using the **-s** flag.

For a “dry-run,” the following arguments are accepted:

-pr

Performs a pre-deploy environment check.

-po

Performs a post-deploy environment check.

To suppress the prompt for the encryption secret and pass it in on the command line instead, issue the **-s** flag.

upgrade

Upgrades all components of an existing GPFS cluster. This command can still be used even if all protocols are not enabled. If a protocol is not enabled, then the packages will still be upgraded, but the service won't be started.

The **spectrumscale** installation toolkit will use environment details in the cluster definition file to perform these tasks. To perform environment health checks prior to and after the upgrade run the **spectrumscale upgrade** command using the **-pr** and **-po** arguments. This is not required, however, because **upgrade** with no arguments will also run this. The following arguments are accepted:

-ve

shows the current versions of installed packages and the available version to upgrade to

-pr

performs health checks on the cluster prior to the upgrade

-po

performs health checks on the cluster after the upgrade has been completed

Exit status

0 Successful completion.

nonzero

A failure has occurred.

Security

You must have root authority to run the **spectrumscale** command.

The node on which the command is issued must be able to execute remote shell commands on any other node in the cluster without the use of a password and without producing any extraneous messages. See the following *IBM Spectrum Scale: Administration and Programming Reference* topic: “Requirements for administering a GPFS file system” on page 1.

Examples

1. To instantiate your chef zero server, issue a command similar to the following:
`spectrumscale setup -s 192.168.0.1`
2. To designate protocol nodes in your environment to use for the installation, issue this command:
`./spectrumscale node add FQDN -p AccessIP`

3. To enable NFS on all protocol nodes, issue this command:
`./spectrumscale enable nfs`
4. To install IBM Spectrum Scale on your defined environment, issue this command:
`./spectrumscale install`
5. To deploy protocols on your defined environment, issue this command:
`./spectrumscale deploy`
6. To delete a node from the configuration, issue this command:
`./spectrumscale node delete FQDN`
7. To enable File Authentication with AD server on all protocol nodes, issue this command:
`./spectrumscale auth file ad`
8. To deploy protocols on your defined environment, using Mysecretkey to access encrypted passwords, issue this command:
`./spectrumscale deploy -s MYsecretkey`
9. To add GPFS Graphical User Interface servers to the cluster definition file, issue this command:
`./spectrumscale node add gpfsnode3 -g`

See also

See also the following *IBM Spectrum Scale: Administration and Programming Reference* topics:

- *Configuring with the spectrumscale installation toolkit* in the *IBM Spectrum Scale: Advanced Administration Guide*.
- “mmchconfig command” on page 331
- “mmlscluster command” on page 524
- “mmlsconfig command” on page 526
- “mmnfs command” on page 570
- “mmobj command” on page 581
- “mmsmb command” on page 663
- “mmuserauth command” on page 690

See also the topic about installing GPFS on Linux nodes in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Location

`/usr/lpp/mmfs/4.2.0.0/installer`

Chapter 12. GPFS programming interfaces

A list of all the GPFS programming interfaces and a short description of each is presented in this topic.

The GPFS APIs are not supported on Windows.

Table 29 summarizes the GPFS programming interfaces.

Table 29. GPFS programming interfaces

Interface	Purpose
"gpfs_acl_t structure" on page 726	Contains buffer mapping for the gpfs_getacl() and gpfs_putacl() subroutines.
"gpfs_clone_copy() subroutine" on page 727	Creates a file clone of a read-only clone parent file.
"gpfs_clone_snap() subroutine" on page 729	Creates a read-only clone parent from a source file.
"gpfs_clone_split() subroutine" on page 731	Splits a file clone from its clone parent.
"gpfs_clone_unsnap() subroutine" on page 733	Changes a clone parent with no file clones back to a regular file.
"gpfs_close_inodescan() subroutine" on page 735	Closes an inode scan.
"gpfs_cmp_fssnapid() subroutine" on page 736	Compares two file system snapshot IDs.
"gpfs_declone() subroutine" on page 738	Removes file clone references to clone parent blocks.
"gpfs_direntx_t structure" on page 740	Contains attributes of a GPFS directory entry.
"gpfs_direntx64_t structure" on page 742	Contains attributes of a GPFS directory entry.
"gpfs_fcntl() subroutine" on page 744	Performs operations on an open file.
"gpfs_fgetattrs() subroutine" on page 747	Retrieves all extended file attributes in opaque format.
"gpfs_fputattrs() subroutine" on page 749	Sets all the extended file attributes for a file.
"gpfs_fputattrswithpathname() subroutine" on page 751	Sets all of the extended file attributes for a file and invokes the policy engine for RESTORE rules.
"gpfs_free_fssnaphandle() subroutine" on page 753	Frees a GPFS file system snapshot handle.
"gpfs_fssnap_handle_t structure" on page 754	Contains a handle for a GPFS file system or snapshot.
"gpfs_fssnap_id_t structure" on page 755	Contains a permanent identifier for a GPFS file system or snapshot.
"gpfs_fstat() subroutine" on page 756	Returns exact file status for a GPFS file.
"gpfs_get_fsname_from_fssnaphandle() subroutine" on page 758	Obtains a file system name from its snapshot handle.
"gpfs_get_fssnaphandle_by_fssnapid() subroutine" on page 759	Obtains a file system snapshot handle using its snapshot ID.
"gpfs_get_fssnaphandle_by_name() subroutine" on page 761	Obtains a file system snapshot handle using its name.
"gpfs_get_fssnaphandle_by_path() subroutine" on page 763	Obtains a file system snapshot handle using its path name.
"gpfs_get_fssnapid_from_fssnaphandle() subroutine" on page 765	Obtains a file system snapshot ID using its handle.
"gpfs_get_pathname_from_fssnaphandle() subroutine" on page 767	Obtains a file system path name using its snapshot handle.
"gpfs_get_snapdirname() subroutine" on page 769	Obtains the name of the directory containing global snapshots.

Table 29. GPFS programming interfaces (continued)

Interface	Purpose
"gpfs_get_snapname_from_fssnaphandle() subroutine" on page 771	Obtains a snapshot name using its file system snapshot handle.
"gpfs_getacl() subroutine" on page 773	Retrieves the access control information for a GPFS file.
"gpfs_iattr_t structure" on page 775	Contains attributes of a GPFS inode.
"gpfs_iattr64_t structure" on page 778	Contains attributes of a GPFS inode.
"gpfs_icolse() subroutine" on page 782	Closes a file given its inode file handle.
"gpfs_ifile_t structure" on page 784	Contains a handle for a GPFS inode.
"gpfs_igetattrs() subroutine" on page 785	Retrieves extended file attributes in opaque format.
"gpfs_igetattrsx() subroutine" on page 787	Retrieves extended file attributes; provides an option to include DMAPI attributes.
"gpfs_igetfilesetname() subroutine" on page 789	Returns the name of the fileset defined by a fileset ID.
"gpfs_igetstoragepool() subroutine" on page 791	Returns the name of the storage pool for the given storage pool ID.
"gpfs_iopen() subroutine" on page 793	Opens a file or directory by inode number.
"gpfs_iopen64() subroutine" on page 795	Opens a file or directory by inode number.
"gpfs_iputattrsx() subroutine" on page 797	Sets the extended file attributes for a file.
"gpfs_iread() subroutine" on page 800	Reads a file opened by gpfs_iopen() .
"gpfs_ireaddir() subroutine" on page 802	Reads the next directory entry.
"gpfs_ireaddir64() subroutine" on page 804	Reads the next directory entry.
"gpfs_ireadlink() subroutine" on page 806	Reads a symbolic link by inode number.
"gpfs_ireadlink64() subroutine" on page 808	Reads a symbolic link by inode number.
"gpfs_ireadx() subroutine" on page 810	Performs block level incremental read of a file within an incremental inode scan.
"gpfs_iscan_t structure" on page 813	Contains a handle for an inode scan of a GPFS file system or snapshot.
"gpfs_lib_init() subroutine" on page 814	Sets up a GPFS interface for additional calls.
"gpfs_lib_term() subroutine" on page 815	Cleans up after GPFS interface calls have been completed.
"gpfs_next_inode() subroutine" on page 816	Retrieves the next inode from the inode scan.
"gpfs_next_inode64() subroutine" on page 818	Retrieves the next inode from the inode scan.
"gpfs_next_inode_with_xattrs() subroutine" on page 820	Retrieves the next inode and its extended attributes from the inode scan.
"gpfs_next_inode_with_xattrs64() subroutine" on page 822	Retrieves the next inode and its extended attributes from the inode scan.
"gpfs_next_xattr() subroutine" on page 824	Returns individual attributes and their values.
"gpfs_opaque_acl_t structure" on page 826	Contains buffer mapping for the gpfs_getacl() and gpfs_putacl() subroutines.
"gpfs_open_inodescan() subroutine" on page 827	Opens an inode scan of a file system or snapshot.
"gpfs_open_inodescan64() subroutine" on page 830	Opens an inode scan of a file system or snapshot.
"gpfs_open_inodescan_with_xattrs() subroutine" on page 833	Opens an inode file and extended attributes for an inode scan.
"gpfs_open_inodescan_with_xattrs64() subroutine" on page 836	Opens an inode file and extended attributes for an inode scan.
"gpfs_prealloc() subroutine" on page 839	Pre-allocates disk storage for a GPFS file.

Table 29. GPFS programming interfaces (continued)

Interface	Purpose
"gpfs_putacl() subroutine" on page 841	Restores the access control information for a GPFS file.
"gpfs_quotactl() subroutine" on page 843	Manipulates disk quotas on file systems.
"gpfs_quotaInfo_t structure" on page 846	Contains buffer mapping for the gpfs_quotactl() subroutine.
"gpfs_seek_inode() subroutine" on page 848	Advances an inode scan to the specified inode number.
"gpfs_seek_inode64() subroutine" on page 850	Advances an inode scan to the specified inode number.
"gpfs_stat() subroutine" on page 852	Returns exact file status for a GPFS file.
"gpfs_stat_inode() subroutine" on page 854	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode64() subroutine" on page 856	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode_with_xattrs() subroutine" on page 858	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode_with_xattrs64() subroutine" on page 860	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfsFcntlHeader_t structure" on page 862	Contains declaration information for the gpfs_fcntl() subroutine.
"gpfsGetFilesetName_t structure" on page 863	Obtains the fileset name of a file.
"gpfsGetReplication_t structure" on page 864	Obtains the replication factors of a file.
"gpfsGetSetXAttr_t structure" on page 866	Obtains or sets extended attribute values.
"gpfsGetSnapshotName_t structure" on page 868	Obtains the snapshot name of a file.
"gpfsGetStoragePool_t structure" on page 869	Obtains the storage pool name of a file.
"gpfsListXAttr_t structure" on page 870	Lists extended attributes.
"gpfsRestripeData_t structure" on page 871	Restripes the data blocks of a file.
"gpfsSetReplication_t structure" on page 873	Sets the replication factors of a file.
"gpfsSetStoragePool_t structure" on page 875	Sets the assigned storage pool of a file.

gpfs_acl_t

gpfs_acl_t structure

Contains buffer mapping for the `gpfs_getacl()` and `gpfs_putacl()` subroutines.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_acl
{
    gpfs_aclLen_t    acl_len;    /* Total length of this ACL in bytes */
    gpfs_aclLevel_t  acl_level;  /* Reserved (must be zero) */
    gpfs_aclVersion_t  acl_version; /* POSIX or NFS4 ACL */
    gpfs_aclType_t    acl_type;   /* Access, Default, or NFS4 */
    gpfs_aclCount_t   acl_nace;   /* Number of Entries that follow */
    union
    {
        gpfs_ace_v1_t  ace_v1[1]; /* when GPFS_ACL_VERSION_POSIX */
        gpfs_ace_v4_t  ace_v4[1]; /* when GPFS_ACL_VERSION_NFS4 */
        v4Level1_t     v4Level1;  /* when GPFS_ACL_LEVEL_V4FLAGS */
    };
} gpfs_acl_t;
```

Description

The `gpfs_acl_t` structure contains size, version, and ACL type information for the `gpfs_getacl()` and `gpfs_putacl()` subroutines.

Members

acl_len

The total length (in bytes) of this `gpfs_acl_t` structure.

acl_level

Reserved for future use. Currently must be zero.

acl_version

This field contains the version of the GPFS ACL. GPFS supports the following ACL versions: `GPFS_ACL_VERSION_POSIX` and `GPFS_ACL_VERSION_NFS4`. On input to the `gpfs_getacl()` subroutine, set this field to zero.

acl_type

On input to the `gpfs_getacl()` subroutine, set this field to:

- Either `GPFS_ACL_TYPE_ACCESS` or `GPFS_ACL_TYPE_DEFAULT` for POSIX ACLs
- `GPFS_ACL_TYPE_NFS4` for NFS ACLs.

These constants are defined in the `gpfs.h` header file.

acl_nace

The number of ACL entries that are in the array (`ace_v1` or `ace_v4`).

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_clone_copy() subroutine

Creates a file clone of a read-only clone parent file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_clone_copy(const char *sourcePathP, const char *destPathP);
```

Description

The `gpfs_clone_copy()` subroutine creates a writeable file clone from a read-only clone parent file.

Parameters

sourcePathP

The path of a read-only source file to clone. The source file can be a file in a snapshot or a clone parent file created with the `gpfs_clone_snap()` subroutine.

destPathP

The path of the destination file to create. The destination file will become the file clone.

Exit status

If the `gpfs_clone_copy()` subroutine is successful, it returns a value of 0 and creates a file clone from the clone parent.

If the `gpfs_clone_copy()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCESS

Permission denied when writing to the destination path or reading from the source path.

EEXIST

The destination file already exists.

EFAULT

The input argument points outside the accessible address space.

EINVAL

The source or destination does not refer to a regular file or a GPFS file system.

EISDIR

The specified destination file is a directory.

ENAMETOOLONG

The source or destination path name is too long.

gpfs_clone_copy()

ENOENT

The source file does not exist.

ENOSPC

The file system has run out of disk space.

ENOSYS

The **gpfs_clone_copy()** subroutine is not available.

EPERM

The source file is a directory or is not a regular file.

EXDEV

The source file and destination file are not in the same file system.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

Related reference:

“gpfs_clone_snap() subroutine” on page 729

Creates a read-only clone parent from a source file.

“gpfs_clone_split() subroutine” on page 731

Splits a file clone from its clone parent.

“gpfs_clone_unsnap() subroutine” on page 733

Changes a clone parent with no file clones back to a regular file.

gpfs_clone_snap() subroutine

Creates a read-only clone parent from a source file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_clone_snap(const char *sourcePathP, const char *destPathP);
```

Description

The `gpfs_clone_snap()` subroutine creates a read-only clone parent from a source file.

Parameters

sourcePathP

The path of the source file to clone.

destPathP

The path of the destination file to create. The destination file will become a read-only clone parent file.

If `destPathP` is `NULL`, then the source file will be changed in place into a read-only clone parent. When using this method to create a clone parent, the specified file cannot be open for writing or have hard links.

Exit status

If the `gpfs_clone_snap()` subroutine is successful, it returns a value of 0 and creates a read-only clone parent from the source file.

If the `gpfs_clone_snap()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCESS

Permission denied when writing to the destination path or reading from the source path.

EEXIST

The destination file already exists.

EFAULT

The input argument points outside accessible address space.

EINVAL

The source or destination does not refer to a regular file or a GPFS file system.

EISDIR

The specified destination file is a directory.

gpfs_clone_snap()

ENAMETOOLONG

The source or destination path name is too long.

ENOENT

The source file does not exist.

ENOSPC

The file system has run out of disk space.

ENOSYS

The **gpfs_clone_snap()** subroutine is not available.

EPERM

The source file is a directory or is not a regular file, or you tried to create a clone file with depth greater than 1000.

EXDEV

The source file and destination file are not in the same file system.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

Related reference:

“gpfs_clone_copy() subroutine” on page 727

Creates a file clone of a read-only clone parent file.

“gpfs_clone_split() subroutine” on page 731

Splits a file clone from its clone parent.

“gpfs_clone_unsnap() subroutine” on page 733

Changes a clone parent with no file clones back to a regular file.

gpfs_clone_split() subroutine

Splits a file clone from its clone parent.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_clone_split(gpfs_file_t fileDesc, int ancLimit);
```

Description

The `gpfs_clone_split()` subroutine splits a file clone from its clone parent. The `gpfs_declone()` subroutine must be called first to remove all references to the clone parent.

Parameters

fileDesc

File descriptor for the file clone to split from its clone parent.

ancLimit

The ancestor limit specified with one of these values:

GPFS_CLONE_ALL

Remove references to all clone parents.

GPFS_CLONE_PARENT_ONLY

Remove references from the immediate clone parent only.

Exit status

If the `gpfs_clone_split()` subroutine is successful, it returns a value of 0.

If the `gpfs_clone_split()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCESS

Permission denied when writing to the target file.

EBADF

The file descriptor is not valid or is not a GPFS file.

EINVAL

An argument to the function was not valid.

ENOSYS

The `gpfs_clone_split()` subroutine is not available.

EPERM

The file descriptor does not refer to a regular file or a file clone.

gpfs_clone_split()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

Related reference:

“gpfs_clone_copy() subroutine” on page 727

Creates a file clone of a read-only clone parent file.

“gpfs_clone_snap() subroutine” on page 729

Creates a read-only clone parent from a source file.

“gpfs_clone_unsnap() subroutine” on page 733

Changes a clone parent with no file clones back to a regular file.

gpfs_clone_unsnap() subroutine

Changes a clone parent with no file clones back to a regular file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_clone_unsnap(gpfs_file_t fileDesc);
```

Description

The `gpfs_clone_unsnap()` subroutine changes a clone parent with no file clones back to a regular file.

Parameters

`fileDesc`

File descriptor for the clone parent to convert back to a regular file.

Exit status

If the `gpfs_clone_unsnap()` subroutine is successful, it returns a value of 0.

If the `gpfs_clone_unsnap()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCESS

Permission denied when writing to the target file.

EBADF

The file descriptor is not valid or is not a GPFS file.

EINVAL

An argument to the function was not valid.

ENOSYS

The `gpfs_clone_unsnap()` subroutine is not available.

EPERM

The file descriptor does not refer to a regular file or a clone parent.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

Related reference:

gpfs_clone_unsnap()

“gpfs_clone_copy() subroutine” on page 727

Creates a file clone of a read-only clone parent file.

“gpfs_clone_snap() subroutine” on page 729

Creates a read-only clone parent from a source file.

“gpfs_clone_split() subroutine” on page 731

Splits a file clone from its clone parent.

gpfs_close_inodescan() subroutine

Closes an inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
void gpfs_close_inodescan(gpfs_iscan_t *iscan);
```

Description

The **gpfs_close_inodescan()** subroutine closes the scan of the inodes in a file system or snapshot that was opened with the **gpfs_open_inodescan()** subroutine. The **gpfs_close_inodescan()** subroutine frees all storage used for the inode scan and invalidates the **iscan** handle.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

Pointer to the inode scan handle.

Exit status

The **gpfs_close_inodescan()** subroutine returns void.

Exceptions

None.

Error status

None.

Examples

For an example using **gpfs_close_inodescan()**, see `/usr/lpp/mmfs/samples/util/tsgetusage.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_cmp_fssnapid()

gpfs_cmp_fssnapid() subroutine

Compares two file system snapshot IDs.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_cmp_fssnapid(const gpfs_fssnap_id_t *fssnapId1,
                    const gpfs_fssnap_id_t *fssnapId2,
                    int *result);
```

Description

The **gpfs_cmp_fssnapid()** subroutine compares two snapshot IDs for the same file system to determine the order in which the two snapshots were taken. The **result** parameter is set as follows:

- **result** less than zero indicates that snapshot 1 was taken before snapshot 2.
- **result** equal to zero indicates that snapshot 1 and 2 are the same.
- **result** greater than zero indicates that snapshot 1 was taken after snapshot 2.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapId1

File system snapshot ID of the first snapshot.

fssnapId2

File system snapshot ID of the second snapshot.

result

Pointer to an integer indicating the outcome of the comparison.

Exit status

If the **gpfs_cmp_fssnapid()** subroutine is successful, it returns a value of 0 and the **result** parameter is set.

If the **gpfs_cmp_fssnapid()** subroutine is unsuccessful, it returns a value of -1 and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The two snapshots cannot be compared because they were taken from two different file systems.

ENOSYS

The `gpfs_cmp_fssnapid()` subroutine is not available.

GPFS_E_INVALID_FSSNAPID

`fssnapId1` or `fssnapId2` is not a valid snapshot ID.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_declone()

gpfs_declone() subroutine

Removes file clone references to clone parent blocks.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_declone(gpfs_file_t fileDesc, int ancLimit, gpfs_off64_t nBlocks,
                gpfs_off64_t *offsetP);
```

Description

The **gpfs_declone()** subroutine removes all file clone references to a clone parent by copying the clone parent blocks to the file clone.

Parameters

fileDesc

The file descriptor for the file clone.

ancLimit

The ancestor limit specified with one of these values:

GPFS_CLONE_ALL

Remove references to all clone parents.

GPFS_CLONE_PARENT_ONLY

Remove references from the immediate clone parent only.

nBlocks

The maximum number of GPFS blocks to copy.

offsetP

A pointer to the starting offset within the file clone. This pointer will be updated to the offset of the next block to process or -1 if there no more blocks.

Exit status

If the **gpfs_declone()** subroutine is successful, it returns a value of 0.

If the **gpfs_declone()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCESS

Permission denied when writing to the target file.

EBADF

The file descriptor is not valid or is not a GPFS file.

EFAULT

The input argument points outside the accessible address space.

EINVAL

An argument to the function was not valid.

ENOSPC

The file system has run out of disk space.

ENOSYS

The **gpfs_declone()** subroutine is not available.

EPERM

The file descriptor does not refer to a regular file.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_direntx_t

gpfs_direntx_t structure

Contains attributes of a GPFS directory entry.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_direntx
{
    int          d_version;      /* this struct's version */
    unsigned short d_reclen;    /* actual size of this struct including
                                null terminated variable length d_name */
    unsigned short d_type;      /* Types are defined below */
    gpfs_ino_t    d_ino;        /* File inode number */
    gpfs_gen_t    d_gen;        /* Generation number for the inode */
    char          d_name[256];  /* null terminated variable length name */
} gpfs_direntx_t;

/* File types for d_type field in gpfs_direntx_t */
#define GPFS_DE_OTHER    0
#define GPFS_DE_DIR     4
#define GPFS_DE_REG     8
#define GPFS_DE_LNK    10
#define GPFS_DE_DEL    16
```

Description

The `gpfs_direntx_t` structure contains the attributes of a GPFS directory entry.

Members

`d_version`

The version number of this structure.

`d_reclen`

The actual size of this structure including the null-terminated variable-length `d_name` field.

To allow some degree of forward compatibility, careful callers should use the `d_reclen` field for the size of the structure rather than the `sizeof()` function.

`d_type`

The type of directory.

`d_ino`

The directory inode number.

`d_gen`

The directory generation number.

`d_name`

Null-terminated variable-length name of the directory.

Examples

For an example using `gpfs_direntx_t`, see `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_direntx64_t structure

Contains attributes of a GPFS directory entry.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_direntx64
{
    int            d_version;        /* this struct's version */
    unsigned short d_reclen;        /* actual size of this struct including
                                     null terminated variable length d_name */
    unsigned short d_type;          /* Types are defined below */
    gpfs_ino64_t  d_ino;            /* File inode number */
    gpfs_gen64_t  d_gen;            /* Generation number for the inode */
    unsigned int  d_flags;          /* Flags are defined below */
    char          d_name[1028];     /* null terminated variable length name */
                                     /* (1020+null+7 byte pad to double word) */
                                     /* to handle up to 255 UTF-8 chars */
} gpfs_direntx64_t;

/* File types for d_type field in gpfs_direntx64_t */
#define GPFS_DE_OTHER    0
#define GPFS_DE_DIR     4
#define GPFS_DE_REG     8
#define GPFS_DE_LNK    10
#define GPFS_DE_DEL    16

/* Define flags for gpfs_direntx64_t */
#define GPFS_DEFLAG_NONE    0x0000 /* Default value, no flags set */
#define GPFS_DEFLAG_JUNCTION 0x0001 /* DirEnt is a fileset junction */
#define GPFS_DEFLAG_IJUNCTION 0x0002 /* DirEnt is a inode space junction */
#define GPFS_DEFLAG_ORPHAN   0x0004 /* DirEnt is an orphan (pcache) */
```

Description

The `gpfs_direntx64_t` structure contains the attributes of a GPFS directory entry.

Members

`d_version`

The version number of this structure.

`d_reclen`

The actual size of this structure including the null-terminated variable-length `d_name` field.

To allow some degree of forward compatibility, careful callers should use the `d_reclen` field for the size of the structure rather than the `sizeof()` function.

`d_type`

The type of directory.

`d_ino`

The directory inode number.

`d_gen`

The directory generation number.

`d_flags`

The directory flags.

d_name

Null-terminated variable-length name of the directory.

Examples

See the `gpfs_direntx_t` example in `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_fcntl()

gpfs_fcntl() subroutine

Performs operations on an open file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_fcntl(gpfs_file_t fileDesc, void* fcntlArgP);
```

Description

The **gpfs_fcntl()** subroutine is used to pass file access pattern information and to control certain file attributes on behalf of an open file. More than one operation can be requested with a single invocation of **gpfs_fcntl()**. The type and number of operations is determined by the second operand, **fcntlArgP**, which is a pointer to a data structure built in memory by the application. This data structure consists of:

- A fixed length header, mapped by **gpfsFcntlHeader_t**.
- A variable list of individual file access hints, directives or other control structures:
 - File access hints:
 - **gpfsAccessRange_t**
 - **gpfsFreeRange_t**
 - **gpfsMultipleAccessRange_t**
 - **gpfsClearFileCache_t**
 - File access directives:
 - **gpfsCancelHints_t**
 - **gpfsDataShipMap_t**
 - **gpfsDataShipStart_t**
 - **gpfsDataShipStop_t**
 - Platform-independent extended attribute operations:
 - **gpfsGetSetXAttr_t**
 - **gpfsListXAttr_t**
 - Other file attribute operations:
 - **gpfsGetFilesetName_t**
 - **gpfsGetReplication_t**
 - **gpfsGetSnapshotName_t**
 - **gpfsGetStoragePool_t**
 - **gpfsRestripeData_t**
 - **gpfsSetReplication_t**
 - **gpfsSetStoragePool_t**

These hints, directives and other operations may be mixed within a single **gpfs_fcntl()** subroutine, and are performed in the order that they appear. A subsequent hint or directive may cancel out a preceding one.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

The file descriptor identifying the file to which GPFS applies the hints and directives.

fcntlArgP

A pointer to the list of operations to be passed to GPFS.

Exit status

If the `gpfs_fcntl()` subroutine is successful, it returns a value of 0.

If the `gpfs_fcntl()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EBADF

The file descriptor is not valid.

EINVAL

The file descriptor does not refer to a GPFS file or a regular file.

The system call is not valid.

ENOSYS

The `gpfs_fcntl()` subroutine is not supported under the current file system format.

Examples

1. This programming segment releases all cache data held by the file *handle* and tell GPFS that the subroutine will write the portion of the file with file offsets between 2 GB and 3 GB minus one:

```
struct
{
    gpfsFcntlHeader_t hdr;
    gpfsClearFileCache_t rel;
    gpfsAccessRange_t acc;
} arg;

arg.hdr.totalLength = sizeof(arg);
arg.hdr.fcntlVersion = GPFS_FCNTL_CURRENT_VERSION;
arg.hdr.fcntlReserved = 0;
arg.rel.structLen = sizeof(arg.rel);
arg.rel.structType = GPFS_CLEAR_FILE_CACHE;
arg.acc.structLen = sizeof(arg.acc);
arg.acc.structType = GPFS_ACCESS_RANGE;
arg.acc.start = 2LL * 1024LL * 1024LL * 1024LL;
arg.acc.length = 1024 * 1024 * 1024;
arg.acc.isWrite = 1;
rc = gpfs_fcntl(handle, &arg);
```

2. This programming segment gets the storage pool name and fileset name of a file from GPFS.

```
struct {
    gpfsFcntlHeader_t hdr;
    gpfsGetStoragePool_t pool;
    gpfsGetFilesetName_t fileset;
} fcntlArg;
fcntlArg.hdr.totalLength = sizeof(fcntlArg.hdr) + sizeof(fcntlArg.pool) + sizeof(fcntlArg.fileset);
```

gpfs_fcntl()

```
fcntlArg.hdr.fcntlVersion = GPFS_FCNTL_CURRENT_VERSION;
fcntlArg.hdr.fcntlReserved = 0;

fcntlArg.pool.structLen = sizeof(fcntlArg.pool);
fcntlArg.pool.structType = GPFS_FCNTL_GET_STORAGEPOOL;

fcntlArg.fileset.structLen = sizeof(fcntlArg.fileset);
fcntlArg.fileset.structType = GPFS_FCNTL_GET_FILESETNAME;

rc = gpfs_fcntl(fd, &fcntlArg);
```

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_fgetattrs() subroutine

Retrieves all extended file attributes in opaque format.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_fgetattrs(gpfs_file_t fileDesc,
                  int flags,
                  void *bufferP,
                  int bufferSize,
                  int *attrSizeP);
```

Description

The `gpfs_fgetattrs()` subroutine, together with `gpfs_fputattrs()`, is intended for use by a backup program to save (`gpfs_fgetattrs()`) and restore (`gpfs_fputattrs()`) extended file attributes such as ACLs, DMAPI attributes, and other information for the file. If the file has no extended attributes, the `gpfs_fgetattrs()` subroutine returns a value of 0, but sets `attrSizeP` to 0 and leaves the contents of the buffer unchanged.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

The file descriptor identifying the file whose extended attributes are being retrieved.

flags

Must have one of the following values:

GPFS_ATTRFLAG_DEFAULT

Saves the attributes for file placement and the currently assigned storage pool.

GPFS_ATTRFLAG_NO_PLACEMENT

Does not save attributes for file placement or the currently assigned storage pool.

GPFS_ATTRFLAG_IGNORE_POOL

Saves attributes for file placement but does not save the currently assigned storage pool.

GPFS_ATTRFLAG_USE_POLICY

Uses the restore policy rules to determine the pool ID.

GPFS_ATTRFLAG_INCL_DMAPI

Includes the DMAPI attributes.

GPFS_ATTRFLAG_FINALIZE_ATTRS

Finalizes immutability attributes.

GPFS_ATTRFLAG_SKIP_IMMUTABLE

Skips immutable attributes.

GPFS_ATTRFLAG_INCL_ENCR

Includes encryption attributes.

GPFS_ATTRFLAG_SKIP_CLONE

Skips clone attributes.

gpfs_fgetattrs()

GPFS_ATTRFLAG_MODIFY_CLONEPARENT

Allows modification on the clone parent.

bufferP

Pointer to a buffer to store the extended attribute information.

bufferSize

The size of the buffer that was passed in.

attrSizeP

If successful, returns the actual size of the attribute information that was stored in the buffer. If the **bufferSize** was too small, returns the minimum buffer size.

Exit status

If the **gpfs_fgetattrs()** subroutine is successful, it returns a value of 0.

If the **gpfs_fgetattrs()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EBADF

The file descriptor is not valid.

EFAULT

The address is not valid.

EINVAL

The file descriptor does not refer to a GPFS file.

ENOSPC

bufferSize is too small to return all of the attributes. On return, **attrSizeP** is set to the required size.

ENOSYS

The **gpfs_fgetattrs()** subroutine is not supported under the current file system format.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_fputattrs() subroutine

Sets all the extended file attributes for a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_fputattrs(gpfs_file_t fileDesc,
                  int flags,
                  void *bufferP);
```

Description

The **gpfs_fputattrs()** subroutine, together with **gpfs_fgetattrs()**, is intended for use by a backup program to save (**gpfs_fgetattrs()**) and restore (**gpfs_fputattrs()**) all of the extended attributes of a file. This subroutine also sets the storage pool for the file and sets data replication to the values that are saved in the extended attributes.

If the saved storage pool is not valid or if the **GPFS_ATTRFLAG_IGNORE_POOL** flag is set, GPFS will select the storage pool by matching a **PLACEMENT** rule using the saved file attributes. If GPFS fails to match a placement rule or if there are no placement rules installed, GPFS assigns the file to the system storage pool.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

The file descriptor identifying the file whose extended attributes are being set.

flags

Must have one of the following values:

GPFS_ATTRFLAG_DEFAULT

Restores the previously assigned storage pool and previously assigned data replication.

GPFS_ATTRFLAG_NO_PLACEMENT

Does not change storage pool and data replication.

GPFS_ATTRFLAG_IGNORE_POOL

Selects storage pool and data replication by matching the saved attributes to a placement rule instead of restoring the saved storage pool.

GPFS_ATTRFLAG_USE_POLICY

Uses the restore policy rules to determine the pool ID.

GPFS_ATTRFLAG_INCL_DMAPI

Includes the DMAPI attributes.

GPFS_ATTRFLAG_FINALIZE_ATTRS

Finalizes immutability attributes.

GPFS_ATTRFLAG_SKIP_IMMUTABLE

Skips immutable attributes.

gpfs_fputattrs()

GPFS_ATTRFLAG_INCL_ENCR

Includes encryption attributes.

GPFS_ATTRFLAG_SKIP_CLONE

Skips clone attributes.

GPFS_ATTRFLAG_MODIFY_CLONEPARENT

Allows modification on the clone parent.

Non-placement attributes such as ACLs are always restored, regardless of value of the flag.

bufferP

A pointer to the buffer containing the extended attributes for the file.

If you specify a value of NULL, all extended ACLs for the file are deleted.

Exit status

If the `gpfs_fputattrs()` subroutine is successful, it returns a value of 0.

If the `gpfs_fputattrs()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EBADF

The file descriptor is not valid.

EINVAL

The buffer pointed to by `bufferP` does not contain valid attribute data, or the file descriptor does not refer to a GPFS file.

ENOSYS

The `gpfs_fputattrs()` subroutine is not supported under the current file system format.

Examples

To copy extended file attributes from file `f1` to file `f2`:

```
char buf[4096];
int f1, f2, attrSize, rc;

rc = gpfs_fgetattrs(f1, GPFS_ATTRFLAG_DEFAULT, buf, sizeof(buf), &attrSize);
if (rc != 0)
    ... // error handling
if (attrSize != 0)
    rc = gpfs_fputattrs(f2, 0, buf); // copy attributes from f1 to f2
else
    rc = gpfs_fputattrs(f2, 0, NULL); // f1 has no attributes
// delete attributes on f2
```

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_fputattrswithpathname() subroutine

Sets all of the extended file attributes for a file and invokes the policy engine for **RESTORE** rules.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_fputattrswithpathname(gpfs_file_t fileDesc,
                               int flags,
                               void *bufferP,
                               const char *pathName);
```

Description

The **gpfs_fputattrswithpathname()** subroutine sets all of the extended attributes of a file. In addition, **gpfs_fputattrswithpathname()** invokes the policy engine using the saved attributes to match a **RESTORE** rule to set the storage pool and the data replication for the file. The caller should include the full path to the file (including the file name) to allow rule selection based on file name or path. If the file fails to match a **RESTORE** rule or if there are no **RESTORE** rules installed, GPFS selects the storage pool and data replication as it does when calling **gpfs_fputattrs()**.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from one the following libraries:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

Is the file descriptor that identifies the file whose extended attributes are to be set.

flags

Must have one of the following values:

GPFS_ATTRFLAG_DEFAULT

Uses the saved attributes to match a **RESTORE** rule to set the storage pool and the data replication for the file.

GPFS_ATTRFLAG_NO_PLACEMENT

Does not change storage pool and data replication.

GPFS_ATTRFLAG_IGNORE_POOL

Checks the file to see if it matches a **RESTORE** rule. If the file fails to match a **RESTORE** rule, GPFS ignores the saved storage pool and selects a pool by matching the saved attributes to a **PLACEMENT** rule.

GPFS_ATTRFLAG_USE_POLICY

Uses the restore policy rules to determine the pool ID.

GPFS_ATTRFLAG_INCL_DMAPI

Includes the DMAPI attributes.

GPFS_ATTRFLAG_FINALIZE_ATTRS

Finalizes immutability attributes.

GPFS_ATTRFLAG_SKIP_IMMUTABLE

Skips immutable attributes.

gpfs_fputattrswithpathname()

GPFS_ATTRFLAG_INCL_ENCR

Includes encryption attributes.

GPFS_ATTRFLAG_SKIP_CLONE

Skips clone attributes.

GPFS_ATTRFLAG_MODIFY_CLONEPARENT

Allows modification on the clone parent.

Non-placement attributes such as ACLs are always restored, regardless of value of the flag.

bufferP

A pointer to the buffer containing the extended attributes for the file.

If you specify a value of NULL, all extended ACLs for the file are deleted.

pathName

A pointer to the path name to a file or directory.

Exit status

If the `gpfs_fputattrswithpathname()` subroutine is successful, it returns a value of 0.

If the `gpfs_fputattrswithpathname()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EBADF

The file descriptor is not valid.

EINVAL

The buffer to which `bufferP` points does not contain valid attribute data.

ENOENT

No such file or directory.

ENOSYS

The `gpfs_fputattrswithpathname()` subroutine is not supported under the current file system format.

Examples

Refer to “`gpfs_fputattrs()` subroutine” on page 749 for examples.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_free_fssnaphandle() subroutine

Frees a GPFS file system snapshot handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
void gpfs_free_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle);
```

Description

The **gpfs_free_fssnaphandle()** subroutine frees the snapshot handle that is passed. The return value is always void.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

Exit status

The **gpfs_free_fssnaphandle()** subroutine always returns void.

Exceptions

None.

Error status

None.

Examples

For an example using **gpfs_free_fssnaphandle()**, see `/usr/lpp/mmfs/samples/util/tstimes.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_fssnap_handle_t structure

Contains a handle for a GPFS file system or snapshot.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_fssnap_handle gpfs_fssnap_handle_t;
```

Description

A file system or snapshot is uniquely identified by an **fssnapId** of type **gpfs_fssnap_id_t**. While the **fssnapId** is permanent and global, a shorter **fssnapHandle** is used by the backup application programming interface to identify the file system and snapshot being accessed. The **fssnapHandle**, like a POSIX file descriptor, is volatile and may be used only by the program that created it.

There are three ways to create a file system snapshot handle:

1. By using the name of the file system and snapshot
2. By specifying the path through the mount point
3. By providing an existing file system snapshot ID

Additional subroutines are provided to obtain the permanent, global **fssnapId** from the **fssnapHandle**, or to obtain the path or the names for the file system and snapshot, if they are still available in the file system.

The file system must be mounted in order to use the backup programming application interface. If the **fssnapHandle** is created by the path name, the path may be relative and may specify any file or directory in the file system. Operations on a particular snapshot are indicated with a path to a file or directory within that snapshot. If the **fssnapHandle** is created by name, the file system's unique name may be specified (for example, **fs1**) or its device name may be provided (for example, **/dev/fs1**). To specify an operation on the active file system, the pointer to the snapshot's name should be set to NULL or a zero-length string provided.

The name of the directory under which all snapshots appear may be obtained by the **gpfs_get_snapdirname()** subroutine. By default this is **.snapshots**, but it can be changed using the **mmsnapdir** command. The **gpfs_get_snapdirname()** subroutine returns the currently set value, which is the one that was last set by the **mmsnapdir** command, or the default, if it was never changed.

Members

gpfs_fssnap_handle

File system snapshot handle

Examples

For an example using **gpfs_fssnap_handle_t**, see **/usr/lpp/mmfs/samples/util/tsgetusage.c**.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_fssnap_id_t structure

Contains a permanent identifier for a GPFS file system or snapshot.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_fssnap_id
{
    char opaque[48];
} gpfs_fssnap_id_t;
```

Description

A file system or snapshot is uniquely identified by an **fssnapId** of type **gpfs_fssnap_id_t**. The **fssnapId** is a permanent and global identifier that uniquely identifies an active file system or a read-only snapshot of a file system. Every snapshot of a file system has a unique identifier that is also different from the identifier of the active file system itself.

The **fssnapId** is obtained from an open **fssnapHandle**. Once obtained, the **fssnapId** should be stored along with the file system's data for each backup. The **fssnapId** is required to generate an incremental backup. The **fssnapId** identifies the previously backed up file system or snapshot and allows the inode scan to return only the files and data that have changed since that previous scan.

Members

opaque

A 48 byte area for containing the snapshot identifier.

Examples

For an example using **gpfs_fssnap_id_t**, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_fstat()

gpfs_fstat() subroutine

Returns exact file status for a GPFS file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_fstat(gpfs_file_t fileDesc,
               gpfs_stat64_t *buffer);
```

Description

The **gpfs_fstat()** subroutine is used to obtain exact information about the file associated with the **fileDesc** parameter. This subroutine is provided as an alternative to the **stat()** subroutine, which may not provide exact **mtime** and **atime** values. For more information, see the topic *Exceptions to Open Group technical standards* in the *IBM Spectrum Scale: Advanced Administration Guide*.

read, **write**, or **execute** permission for the named file is not required, but all directories listed in the path leading to the file must be searchable. The file information is written to the area specified by the **buffer** parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

The file descriptor identifying the file for which exact status information is requested.

buffer

A pointer to the **gpfs_stat64_t** structure in which the information is returned. The **gpfs_stat64_t** structure is described in the `sys/stat.h` file.

Exit status

If the **gpfs_fstat()** subroutine is successful, it returns a value of 0.

If the **gpfs_fstat()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EBADF

The file descriptor is not valid.

EINVAL

The file descriptor does not refer to a GPFS file or a regular file.

ENOSYS

The **gpfs_fstat()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_fsnam_from_fssnaphandle()

gpfs_get_fsnam_from_fssnaphandle() subroutine

Obtains a file system name from its snapshot handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
const char *gpfs_get_fsnam_from_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle);
```

Description

The `gpfs_get_fsnam_from_fssnaphandle()` subroutine returns a pointer to the name of file system that is uniquely identified by the file system snapshot handle.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

Exit status

If the `gpfs_get_fsnam_from_fssnaphandle()` subroutine is successful, it returns a pointer to the name of the file system identified by the file system snapshot handle.

If the `gpfs_get_fsnam_from_fssnaphandle()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The `gpfs_get_fsnam_from_fssnaphandle()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_fssnaphandle_by_fssnapid() subroutine

Obtains a file system snapshot handle using its snapshot ID.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_fssnap_handle_t *gpfs_get_fssnaphandle_by_fssnapid(const gpfs_fssnap_id_t *fssnapId);
```

Description

The `gpfs_get_fssnaphandle_by_fssnapid()` subroutine creates a handle for the file system or snapshot that is uniquely identified by the permanent, unique snapshot ID.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapId

File system snapshot ID

Exit status

If the `gpfs_get_fssnaphandle_by_fssnapid()` subroutine is successful, it returns a pointer to the file system snapshot handle.

If the `gpfs_get_fssnaphandle_by_fssnapid()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Space could not be allocated for the file system snapshot handle.

ENOSYS

The `gpfs_get_fssnaphandle_by_fssnapid()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID is not valid.

gpfs_get_fssnaphandle_by_fssnapid()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_fssnaphandle_by_name() subroutine

Obtains a file system snapshot handle using its name.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_fssnap_handle_t *gpfs_get_fssnaphandle_by_name(const char *fsName,
                                                    const char *snapName);
```

Description

The `gpfs_get_fssnaphandle_by_name()` subroutine creates a handle for the file system or snapshot that is uniquely identified by the file system's name and the name of the snapshot.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fsName

A pointer to the name of the file system whose snapshot handle is desired.

snapName

A pointer to the name of the snapshot whose snapshot handle is desired, or NULL to access the active file system rather than a snapshot within the file system.

Exit status

If the `gpfs_get_fssnaphandle_by_name()` subroutine is successful, it returns a pointer to the file system snapshot handle.

If the `gpfs_get_fssnaphandle_by_name()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOENT

The file system name is not valid.

ENOMEM

Space could not be allocated for the file system snapshot handle.

ENOSYS

The `gpfs_get_fssnaphandle_by_name()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

gpfs_get_fssnaphandle_by_name()

GPFS_E_INVALID_SNAPNAME

The snapshot name is not valid.

Examples

For an example using `gpfs_get_fssnaphandle_by_name()`, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_get_fssnaphandle_by_path() subroutine

Obtains a file system snapshot handle using its path name.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_fssnap_handle_t *gpfs_get_fssnaphandle_by_path(const char *pathName);
```

Description

The `gpfs_get_fssnaphandle_by_path()` subroutine creates a handle for the file system or snapshot that is uniquely identified by a path through the file system's mount point to a file or directory within the file system or snapshot.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

pathName

A pointer to the path name to a file or directory within the desired file system or snapshot.

Exit status

If the `gpfs_get_fssnaphandle_by_path()` subroutine is successful, it returns a pointer to the file system snapshot handle.

If the `gpfs_get_fssnaphandle_by_path()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOENT

The path name is not valid.

ENOMEM

Space could not be allocated for the file system snapshot handle.

ENOSYS

The `gpfs_get_fssnaphandle_by_path()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

Examples

For an example using `gpfs_get_fssnaphandle_by_path()`, see `/usr/lpp/mmfs/samples/util/tsgetusage.c`.

gpfs_get_fssnaphandle_by_path()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_fssnapid_from_fssnaphandle() subroutine

Obtains a file system snapshot ID using its handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_get_fssnapid_from_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle,
                                       gpfs_fssnap_id_t *fssnapId);
```

Description

The `gpfs_get_fssnapid_from_fssnaphandle()` subroutine obtains the permanent, globally unique file system snapshot ID of the file system or snapshot identified by the open file system snapshot handle.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

fssnapId

File system snapshot ID.

Exit status

If the `gpfs_get_fssnapid_from_fssnaphandle()` subroutine is successful, it returns a pointer to the file system snapshot ID.

If the `gpfs_get_fssnapid_from_fssnaphandle()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EFAULT

Size mismatch for `fssnapId`.

EINVAL

NULL pointer given for returned `fssnapId`.

ENOSYS

The `gpfs_get_fssnapid_from_fssnaphandle()` subroutine is not available.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

gpfs_get_fssnapid_from_fssnaphandle()

Examples

For an example using `gpfs_get_fssnapid_from_fssnaphandle()`, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_get_pathname_from_fssnaphandle() subroutine

Obtains a file system path name using its snapshot handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
const char *gpfs_get_pathname_from_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle);
```

Description

The `gpfs_get_pathname_from_fssnaphandle()` subroutine obtains the path name of the file system or snapshot identified by the open file system snapshot handle.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

Exit status

If the `gpfs_get_pathname_from_fssnaphandle()` subroutine is successful, it returns a pointer to the path name of the file system or snapshot.

If the `gpfs_get_pathname_from_fssnaphandle()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The `gpfs_get_pathname_from_fssnaphandle()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

Examples

For an example using `gpfs_get_pathname_from_fssnaphandle()`, see `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

gpfs_get_pathname_from_fssnaphandle()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_snapdirname() subroutine

Obtains the name of the directory containing global snapshots.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_get_snapdirname(gpfs_fssnap_handle_t *fssnapHandle,
                        char *snapdirName,
                        int bufLen);
```

Description

The `gpfs_get_snapdirname()` subroutine obtains the name of the directory that contains global snapshots.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

snapdirName

Buffer into which the name of the snapshot directory will be copied.

bufLen

The size of the provided buffer.

Exit status

If the `gpfs_get_snapdirname()` subroutine is successful, it returns a value of 0 and the `snapdirName` and `bufLen` parameters are set.

If the `gpfs_get_snapdirname()` subroutine is unsuccessful, it returns a value of -1 and the global error variable `errno` is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The `gpfs_get_snapdirname()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

gpfs_get_snapdirname()

ERANGE

The buffer is too small to return the snapshot directory name.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSNAPHANDLE

The file system snapshot handle is not valid.

E2BIG The buffer is too small to return the snapshot directory name.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_get_snapname_from_fssnaphandle() subroutine

Obtains a snapshot name using its file system snapshot handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
const char *gpfs_get_snapname_from_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle);
```

Description

The `gpfs_get_snapname_from_fssnaphandle()` subroutine obtains a pointer to the name of a GPFS snapshot given its file system snapshot handle. If the `fssnapHandle` identifies an active file system, as opposed to a snapshot of a file system, `gpfs_get_snapname_from_fssnaphandle()` returns a pointer to a zero-length snapshot name and a successful return code.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

Exit status

If the `gpfs_get_snapname_from_fssnaphandle()` subroutine is successful, it returns a pointer to the name of the snapshot.

If the `gpfs_get_snapname_from_fssnaphandle()` subroutine is unsuccessful, it returns NULL and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The `gpfs_get_snapname_from_fssnaphandle()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

GPFS_E_INVALID_SNAPNAME

The snapshot has been deleted.

gpfs_get_snapname_from_fssnaphandle()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_getacl() subroutine

Retrieves the access control information for a GPFS file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_getacl(const char *pathname,
               int flags,
               void *acl);
```

Description

The **gpfs_getacl()** subroutine, together with the **gpfs_putacl()** subroutine, is intended for use by a backup program to save (**gpfs_getacl()**) and restore (**gpfs_putacl()**) the ACL information for the file.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

pathname

The path identifying the file for which the ACLs are being obtained.

flags

Consists of one of these values:

- 0** Indicates that the **acl** parameter is to be mapped with the **gpfs_opaque_acl_t** structure.

The **gpfs_opaque_acl_t** structure should be used by backup and restore programs.

GPFS_GETACL_STRUCT

Indicates that the **acl** parameter is to be mapped with the **gpfs_acl_t** structure.

The **gpfs_acl_t** structure is provided for applications that need to interpret the ACL.

acl

Pointer to a buffer mapped by the structure **gpfs_opaque_acl_t** or **gpfs_acl_t**, depending on the value of **flags**.

The first four bytes of the buffer must contain its total size.

Exit status

If the **gpfs_getacl()** subroutine is successful, it returns a value of 0.

If the **gpfs_getacl()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

gpfs_getacl()

EINVAL

The path name does not refer to a GPFS file or a regular file.

ENOMEM

Unable to allocate memory for the request.

ENOTDIR

File is not a directory.

ENOSPC

The buffer is too small to return the entire ACL. The required buffer size is returned in the first four bytes of the buffer pointed to by **acl**.

ENOSYS

The **gpfs_getacl()** subroutine is not supported under the current file system format.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_iattr_t structure

Contains attributes of a GPFS inode.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_iattr
{
    int            ia_version;    /* this struct version */
    int            ia_reclen;    /* sizeof this structure */
    int            ia_checksum;  /* validity check on iattr struct */
    gpfs_mode_t   ia_mode;      /* access mode */
    gpfs_uid_t    ia_uid;       /* owner uid */
    gpfs_gid_t    ia_gid;       /* owner gid */
    gpfs_ino_t    ia_inode;     /* file inode number */
    gpfs_gen_t    ia_gen;       /* inode generation number */
    gpfs_nlink_t  ia_nlink;     /* number of links */
    short         ia_flags;     /* Flags (defined below) */
    int           ia_blocksize;  /* preferred block size for io */
    gpfs_mask_t   ia_mask;      /* Initial attribute mask (not used) */
    unsigned int  ia_pad1;      /* reserved space */
    gpfs_off64_t  ia_size;      /* file size in bytes */
    gpfs_off64_t  ia_blocks;    /* 512 byte blocks of disk held by file */
    gpfs_timestruc_t ia_atime;  /* time of last access */
    gpfs_timestruc_t ia_mtime;  /* time of last data modification */
    gpfs_timestruc_t ia_ctime;  /* time of last status change */
    gpfs_dev_t    ia_rdev;      /* id of device */
    unsigned int  ia_xperm;     /* extended attributes (defined below) */
    unsigned int  ia_modsnapid; /* snapshot id of last modification */
    unsigned int  ia_filesetid; /* fileset ID */
    unsigned int  ia_datapoolid; /* storage pool ID for data */
    unsigned int  ia_pad2;      /* reserved space */
} gpfs_iattr_t;

/* Define flags for inode attributes */
#define GPFS_IAFLAG_SNAPDIR      0x0001 /* (obsolete) */
#define GPFS_IAFLAG_USRQUOTA    0x0002 /* inode is a user quota file */
#define GPFS_IAFLAG_GRPQUOTA    0x0004 /* inode is a group quota file */
#define GPFS_IAFLAG_ERROR       0x0008 /* error reading inode */
/* Define flags for inode replication attributes */
#define GPFS_IAFLAG_FILESET_ROOT 0x0010 /* root dir of a fileset */
#define GPFS_IAFLAG_NO_SNAP_RESTORE 0x0020 /* don't restore from snapshots */
#define GPFS_IAFLAG_FILESETQUOTA 0x0040 /* inode is a fileset quota file */
#define GPFS_IAFLAG_COMANAGED    0x0080 /* file data is co-managed */
#define GPFS_IAFLAG_ILLPLACED    0x0100 /* may not be properly placed */
#define GPFS_IAFLAG_REPLMETA     0x0200 /* metadata replication set */
#define GPFS_IAFLAG_REPLDATA     0x0400 /* data replication set */
#define GPFS_IAFLAG_EXPOSED      0x0800 /* may have data on suspended disks */
#define GPFS_IAFLAG_ILLREPLICATED 0x1000 /* may not be properly replicated */
#define GPFS_IAFLAG_UNBALANCED    0x2000 /* may not be properly balanced */
#define GPFS_IAFLAG_DATAUPDATEMISS 0x4000 /* has stale data blocks on
unavailable disk */
#define GPFS_IAFLAG_METAUPDATEMISS 0x8000 /* has stale metadata on
unavailable disk */

#define GPFS_IAFLAG_IMMUTABLE    0x00010000 /* Immutability */
#define GPFS_IAFLAG_INDEFRETENT  0x00020000 /* Indefinite retention */
#define GPFS_IAFLAG_SECUREDELETE 0x00040000 /* Secure deletion */

#define GPFS_IAFLAG_TRUNCMANAGED 0x00080000 /* dmapi truncate event enabled */
#define GPFS_IAFLAG_READMANAGED  0x00100000 /* dmapi read event enabled */
#define GPFS_IAFLAG_WRITEMANAGED 0x00200000 /* dmapi write event enabled */
```

gpfs_iattr_t

```
#define GPFS_IAFLAG_APPENDONLY    0x00400000 /* AppendOnly only */
#define GPFS_IAFLAG_DELETED      0x00800000 /* inode has been deleted */

/* Define flags for extended attributes */
#define GPFS_IAXPERM_ACL          0x0001 /* file has acls */
#define GPFS_IAXPERM_XATTR       0x0002 /* file has extended attributes */
#define GPFS_IAXPERM_DMATTR      0x0004 /* file has dm attributes */
#define GPFS_IAXPERM_DOSATTR     0x0008 /* file has non-default dos attrs */
#define GPFS_IAXPERM_RPATTR      0x0010 /* file has restore policy attrs */
```

Description

The `gpfs_iattr_t` structure contains the various attributes of a GPFS inode.

Members

`ia_version`

The version number of this structure.

`ia_reclen`

The size of this structure.

`ia_checksum`

The checksum for this `gpfs_iattr` structure.

`ia_mode`

The access mode for this inode.

`ia_uid`

The owner user ID for this inode.

`ia_gid`

The owner group ID for this inode.

`ia_inode`

The file inode number.

`ia_gen`

The inode generation number.

`ia_nlink`

The number of links for this inode.

`ia_flags`

The flags defined for inode attributes.

`ia_blocksize`

The preferred block size for I/O.

`ia_mask`

The initial attribute mask (not used).

`ia_pad1`

Reserved space.

`ia_size`

The file size in bytes.

`ia_blocks`

The number of 512 byte blocks of disk held by the file.

`ia_atime`

The time of last access.

ia_mtime

The time of last data modification.

ia_ctime

The time of last status change.

ia_rdev

The ID of the device.

ia_xperm

Indicator - nonzero if file has extended ACL.

ia_modsnapid

Internal snapshot ID indicating the last time that the file was modified. Internal snapshot IDs for the current snapshots are displayed by the **mmlssnapshot** command.

ia_filesetid

The fileset ID for the inode.

ia_datapoolid

The storage pool ID for data for the inode.

ia_pad2

Reserved space.

Examples

For an example using **gpfs_iattr_t**, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_iattr64_t structure

Contains attributes of a GPFS inode.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_iattr64
{
    int                ia_version;    /* this struct version */
    int                ia_reclen;    /* sizeof this structure */
    int                ia_checksum;  /* validity check on iattr struct */
    gpfs_mode_t       ia_mode;       /* access mode */
    gpfs_uid64_t      ia_uid;        /* owner uid */
    gpfs_gid64_t      ia_gid;        /* owner gid */
    gpfs_ino64_t      ia_inode;      /* file inode number */
    gpfs_gen64_t      ia_gen;        /* inode generation number */
    gpfs_nlink64_t    ia_nlink;      /* number of links */
    gpfs_off64_t      ia_size;       /* file size in bytes */
    gpfs_off64_t      ia_blocks;     /* 512 byte blocks of disk held by file */
    gpfs_timestruc64_t ia_atime;     /* time of last access */
    unsigned int      ia_winflags;   /* window's flags (defined below) */
    unsigned int      ia_pad1;       /* reserved space */
    gpfs_timestruc64_t ia_mtime;     /* time of last data modification */
    unsigned int      ia_flags;      /* flags (defined below) */
    unsigned char     ia_repl_data;   /* data replication factor */
    unsigned char     ia_repl_data_max; /* data replication max factor */
    unsigned char     ia_repl_meta;  /* meta data replication factor */
    unsigned char     ia_repl_meta_max; /* meta data replication max factor */
    gpfs_timestruc64_t ia_ctime;     /* time of last status change */
    int               ia_blocksize;  /* preferred block size for io */
    unsigned int      ia_pad3;       /* reserved space */
    gpfs_timestruc64_t ia_createtime; /* creation time */
    gpfs_mask_t       ia_mask;       /* initial attribute mask (not used) */
    int               ia_pad4;       /* reserved space */
    unsigned int      ia_reserved[GPFS_IA64_RESERVED]; /* reserved space */
    unsigned int      ia_xperm;      /* extended attributes (defined below) */
    gpfs_dev_t        ia_dev;        /* id of device containing file */
    gpfs_dev_t        ia_rdev;       /* device id (if special file) */
    unsigned int      ia_pcacheflags; /* pcache inode bits */
    gpfs_snapid64_t   ia_modsnapid;  /* snapshot id of last modification */
    unsigned int      ia_filesetid;  /* fileset ID */
    unsigned int      ia_datapoolid; /* storage pool ID for data */
    gpfs_ino64_t      ia_inode_space_mask; /* inode space mask of this file system */
                                                /* This value is saved in the iattr structure
                                                during backup and used during restore */
    unsigned int      ia_unused[GPFS_IA64_UNUSED]; /* reserved space */
} gpfs_iattr64_t;

#ifdef GPFS_64BIT_INODES
    #undef GPFS_IA_VERSION
    #define GPFS_IA_VERSION GPFS_IA_VERSION64
    #define gpfs_iattr_t gpfs_iattr64_t
#endif

/* Define flags for inode attributes */
#define GPFS_IAFLAG_SNAPDIR        0x0001 /* (obsolete) */
#define GPFS_IAFLAG_USRQUOTA      0x0002 /* inode is a user quota file */
#define GPFS_IAFLAG_GRPQUOTA      0x0004 /* inode is a group quota file */
#define GPFS_IAFLAG_ERROR         0x0008 /* error reading inode */
/* Define flags for inode replication attributes */
#define GPFS_IAFLAG_FILESET_ROOT  0x0010 /* root dir of a fileset */
```



```

#define GPFS_IAFLAG_NO_SNAP_RESTORE 0x0020 /* don't restore from snapshots */
#define GPFS_IAFLAG_FILESETQUOTA    0x0040 /* inode is a fileset quota file */
#define GPFS_IAFLAG_COMANAGED        0x0080 /* file data is co-managed */
#define GPFS_IAFLAG_ILLPLACED        0x0100 /* may not be properly placed */
#define GPFS_IAFLAG_REPLMETA         0x0200 /* metadata replication set */
#define GPFS_IAFLAG_REPLDATA         0x0400 /* data replication set */
#define GPFS_IAFLAG_EXPOSED          0x0800 /* may have data on suspended disks */
#define GPFS_IAFLAG_ILLREPLICATED    0x1000 /* may not be properly replicated */
#define GPFS_IAFLAG_UNBALANCED       0x2000 /* may not be properly balanced */
#define GPFS_IAFLAG_DATAUPDATEMISS   0x4000 /* has stale data blocks on
unavailable disk */
#define GPFS_IAFLAG_METAUPDATEMISS   0x8000 /* has stale metadata on
unavailable disk */

#define GPFS_IAFLAG_IMMUTABLE         0x00010000 /* Immutability */
#define GPFS_IAFLAG_INDEFRETENT       0x00020000 /* Indefinite retention */
#define GPFS_IAFLAG_SECUREDELETE     0x00040000 /* Secure deletion */

#define GPFS_IAFLAG_TRUNCMANAGED      0x00080000 /* dmapi truncate event enabled */
#define GPFS_IAFLAG_READMANAGED      0x00100000 /* dmapi read event enabled */
#define GPFS_IAFLAG_WRITEMANAGED     0x00200000 /* dmapi write event enabled */

#define GPFS_IAFLAG_APPENDONLY        0x00400000 /* AppendOnly only */
#define GPFS_IAFLAG_DELETED           0x00800000 /* inode has been deleted */

/* Define flags for window's attributes */
#define GPFS_IWINFLAG_ARCHIVE         0x0001 /* Archive */
#define GPFS_IWINFLAG_HIDDEN          0x0002 /* Hidden */
#define GPFS_IWINFLAG_NOTINDEXED      0x0004 /* Not content indexed */
#define GPFS_IWINFLAG_OFFLINE         0x0008 /* Off-line */
#define GPFS_IWINFLAG_READONLY        0x0010 /* Read-only */
#define GPFS_IWINFLAG_REPARSE         0x0020 /* Reparse point */
#define GPFS_IWINFLAG_SYSTEM          0x0040 /* System */
#define GPFS_IWINFLAG_TEMPORARY       0x0080 /* Temporary */
#define GPFS_IWINFLAG_COMPRESSED      0x0100 /* Compressed */
#define GPFS_IWINFLAG_ENCRYPTED        0x0200 /* Encrypted */
#define GPFS_IWINFLAG_SPARSE          0x0400 /* Sparse file */
#define GPFS_IWINFLAG_HASSTREAMS      0x0800 /* Has streams */

/* Define flags for extended attributes */
#define GPFS_IAXPERM_ACL               0x0001 /* file has acls */
#define GPFS_IAXPERM_XATTR            0x0002 /* file has extended attributes */
#define GPFS_IAXPERM_DMATTR           0x0004 /* file has dm attributes */
#define GPFS_IAXPERM_DOSATTR          0x0008 /* file has non-default dos attrs */
#define GPFS_IAXPERM_RPATR            0x0010 /* file has restore policy attrs */

/* Define flags for pcache bits defined in the inode */
#define GPFS_ICAFLAG_CACHED           0x0001 /* "cached complete" */
#define GPFS_ICAFLAG_CREATE           0x0002 /* "created" */
#define GPFS_ICAFLAG_DIRTY            0x0004 /* "data dirty" */
#define GPFS_ICAFLAG_LINK             0x0008 /* "hard linked" */
#define GPFS_ICAFLAG_SETATTR          0x0010 /* "attr changed" */
#define GPFS_ICAFLAG_LOCAL            0x0020 /* "local" */
#define GPFS_ICAFLAG_APPEND           0x0040 /* "append" */
#define GPFS_ICAFLAG_STATE            0x0080 /* "has remote state" */

```

Description

The `gpfs_iattr64_t` structure contains the various attributes of a GPFS inode.

Members

`ia_version`

The version number of this structure.

gpfs_iattr64_t

ia_reclen

The size of this structure.

ia_checksum

The checksum for this **gpfs_iattr64** structure.

ia_mode

The access mode for this inode.

ia_uid

The owner user ID for this inode.

ia_gid

The owner group ID for this inode.

ia_inode

The file inode number.

ia_gen

The inode generation number.

ia_nlink

The number of links for this inode.

ia_size

The file size in bytes.

ia_blocks

The number of 512 byte blocks of disk held by the file.

ia_atime

The time of last access.

ia_winflags

The Windows flags.

ia_pad1

Reserved space.

ia_mtime

The time of last data modification.

ia_flags

The flags defined for inode attributes.

ia_repl_data

The data replication factor.

ia_repl_data_max

The maximum data replication factor.

ia_repl_meta

The metadata replication factor.

ia_repl_meta_max

The maximum metadata replication factor.

ia_ctime

The time of last status change.

ia_blocksize

The preferred block size for I/O.

ia_pad3

Reserved space.

ia_createtime

The creation time.

ia_mask

The initial attribute mask (not used).

ia_pad4

Reserved space.

ia_reserved

Reserved space.

ia_xperm

Indicator - nonzero if file has extended ACL.

ia_dev

The ID of the device containing the file.

ia_rdev

The ID of the device.

ia_pcacheflags

The pcache inode bits.

ia_modsnapid

Internal snapshot ID indicating the last time that the file was modified. Internal snapshot IDs for the current snapshots are displayed by the **mmlssnapshot** command.

ia_filesetid

The fileset ID for the inode.

ia_datapoolid

The storage pool ID for data for the inode.

ia_inode_space_mask

The inode space mask of this file system. This value is saved in the **iattr** structure during backup and used during restore.

ia_unused

Reserved space.

Examples

See the **gpfs_iattr_t** example in `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_icolse()

gpfs_icolse() subroutine

Closes a file given its inode file handle.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
void gpfs_icolse(gpfs_ifile_t *ifile);
```

Description

The **gpfs_icolse()** subroutine closes an open file descriptor created by **gpfs_iopen()**.

For an overview of using **gpfs_icolse()** in a backup application, see the topic *Using APIs to develop backup applications* in the *IBM Spectrum Scale: Advanced Administration Guide*

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

ifile

Pointer to **gpfs_ifile_t** from **gpfs_iopen()**.

Exit status

The **gpfs_icolse()** subroutine returns void.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The **gpfs_icolse()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

Examples

For an example using **gpfs_icolse()**, see `/usr/lpp/mmfs/samples/util/tsreaddir.c`.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_ifile_t

gpfs_ifile_t structure

Contains a handle for a GPFS inode.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_ifile gpfs_ifile_t;
```

Description

The `gpfs_ifile_t` structure contains a handle for the file of a GPFS inode.

Members

`gpfs_ifile`

The handle for the file of a GPFS inode.

Examples

For an example using `gpfs_ifile_t`, see `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_igetattrs() subroutine

Retrieves extended file attributes in opaque format.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_igetattrs(gpfs_ifile_t *ifile,
                  void *buffer,
                  int bufferSize,
                  int *attrSize);
```

Description

The **gpfs_igetattrs()** subroutine retrieves all extended file attributes in opaque format. This subroutine is intended for use by a backup program to save all extended file attributes (ACLs, attributes, and so forth). If the file does not have any extended attributes, the subroutine sets **attrSize** to zero.

Notes:

1. This call does not return extended attributes used for the Data Storage Management (XDSM) API (also known as DMAPI).
2. Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:
 - libgpfs.a for AIX
 - libgpfs.so for Linux

Parameters

ifile

Pointer to **gpfs_ifile_t** from **gpfs_iopen()**.

buffer

Pointer to buffer for returned attributes.

bufferSize

Size of the buffer.

attrSize

Pointer to returned size of attributes.

Exit status

If the **gpfs_igetattrs()** subroutine is successful, it returns a value of 0.

If the **gpfs_igetattrs()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

gpfs_igetattrs()

ENOSPC

The buffer is too small to return all attributes. Field **attrSize** will be set to the size necessary.

ENOSYS

The **gpfs_igetattrs()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_IFILE

Incorrect **ifile** parameters.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_igetattrsx() subroutine

Retrieves extended file attributes; provides an option to include DMAPI attributes.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_igetattrsx(gpfs_ifile_t *ifile,
                   int flags,
                   void *buffer,
                   int bufferSize,
                   int *attrSize);
```

Description

The `gpfs_igetattrsx()` subroutine retrieves all extended file attributes in opaque format. It provides the same function as `gpfs_igetattr()` but includes a `flags` parameter that allows the caller to back up and restore DMAPI attributes.

This function is intended for use by a backup program to save (and restore, using the related subroutine `gpfs_iputattrsx()`) all extended file attributes (ACLs, user attributes, and so forth) in one call. If the file does not have any extended attributes, the subroutine sets `attrSize` to zero.

Notes:

1. This call can optionally return extended attributes used for the Data Storage Management (XDSM) API (also known as DMAPI).
2. Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:
 - libgpfs.a for AIX
 - libgpfs.so for Linux

Parameters

ifile

Pointer to `gpfs_ifile_t` from `gpfs_iopen()`.

flags

Flags must have one of the following values:

GPFS_ATTRFLAG_NO_PLACEMENT

File attributes for placement are not saved, and neither is the current storage pool.

GPFS_ATTRFLAG_IGNORE_PLACEMENT

File attributes for placement are saved, but the current storage pool is not.

GPFS_ATTRFLAG_INCL_DMAPI

File attributes for DMAPI are included in the returned buffer.

GPFS_ATTRFLAG_USE_POLICY

Uses the restore policy rules to determine the pool ID.

GPFS_ATTRFLAG_INCL_DMAPI

Includes the DMAPI attributes.

GPFS_ATTRFLAG_FINALIZE_ATTRS

Finalizes immutability attributes.

gpfs_igetattrsx()

GPFS_ATTRFLAG_SKIP_IMMUTABLE

Skips immutable attributes.

GPFS_ATTRFLAG_INCL_ENCR

Includes encryption attributes.

GPFS_ATTRFLAG_SKIP_CLONE

Skips clone attributes.

GPFS_ATTRFLAG_MODIFY_CLONEPARENT

Allows modification on the clone parent.

buffer

A pointer to the buffer for returned attributes.

bufferSize

Size of the buffer.

attrSize

Pointer to returned size of attributes.

Exit status

If the **gpfs_igetattrsx()** subroutine is successful, it returns a value of 0.

If the **gpfs_igetattrsx()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

Not a GPFS file, or the flags provided are not valid.

ENOSPC

The buffer is too small to return all attributes. Field **attrSize** will be set to the size necessary.

ENOSYS

The **gpfs_igetattrsx()** subroutine is not available.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_igetfilesetname() subroutine

Returns the name of the fileset defined by a fileset ID.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_igetfilesetname(gpfs_iscan_t *iscan,
                        unsigned int filesetId,
                        void *buffer,
                        int bufferSize);
```

Description

The **gpfs_igetfilesetname()** subroutine is part of the backup by inode interface. The caller provides a pointer to the scan descriptor used to obtain the fileset ID. This library routine will return the name of the fileset defined by the fileset ID. The name is the null-terminated string provided by the administrator when the fileset was defined. The maximum string length is **GPFS_MAXNAMLEN**, which is defined in `/usr/lpp/mmfs/include/gpfs.h`.

Notes:

1. This routine is not thread safe. Only one thread at a time is allowed to invoke this routine for the given scan descriptor.
2. Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:
 - libgpfs.a for AIX
 - libgpfs.so for Linux

Parameters

iscan

Pointer to **gpfs_iscan_t** used to obtain the fileset ID.

filesetId

The fileset ID.

buffer

Pointer to buffer for returned attributes.

bufferSize

Size of the buffer.

Exit status

If the **gpfs_igetfilesetname()** subroutine is successful, it returns a value of 0.

If the **gpfs_igetfilesetname()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

gpfs_igetfilesetname()

Error status

Error codes include but are not limited to the following:

E2BIG The buffer is too small to return the fileset name.

EINTR

The call was interrupted. This routine is not thread safe.

EINVAL

The fileset ID is not valid.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_igetfilesetname()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_ISCAN

The **iscan** parameters were not valid.

Examples

This programming segment gets the fileset name based on the given fileset ID. The returned fileset name is stored in **FileSetNameBuffer**, which has a length of **FileSetNameSize**.

```
gpfs_iscan_t *fsInodeScanP;  
gpfs_igetfilesetname(fsInodeScanP,FileSetId, &FileSetNameBuffer,FileSetNameSize);
```

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_igetstoragepool() subroutine

Returns the name of the storage pool for the given storage pool ID.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_igetstoragepool(gpfs_iscan_t *iscan,
                        unsigned int dataPoolId,
                        void *buffer,
                        int bufferSize);
```

Description

The **gpfs_igetstoragepool()** subroutine is part of the backup by inode interface. The caller provides a pointer to the scan descriptor used to obtain the storage pool ID. This routine returns the name of the storage pool for the given storage pool ID. The name is the null-terminated string provided by the administrator when the storage pool was defined. The maximum string length is **GPFS_MAXNAMLEN**, which is defined in `/usr/lpp/mmfs/include/gpfs.h`.

Notes:

1. This routine is not thread safe. Only one thread at a time is allowed to invoke this routine for the given scan descriptor.
2. Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:
 - libgpfs.a for AIX
 - libgpfs.so for Linux

Parameters

iscan

Pointer to **gpfs_iscan_t** used to obtain the storage pool ID.

dataPoolId

The storage pool ID.

buffer

Pointer to buffer for returned attributes.

bufferSize

Size of the buffer.

Exit status

If the **gpfs_igetstoragepool()** subroutine is successful, it returns a value of 0.

If the **gpfs_igetstoragepool()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

gpfs_igetstoragepool()

Error status

Error codes include but are not limited to the following:

E2BIG The buffer is too small to return the storage pool name.

EINTR

The call was interrupted. This routine is not thread safe.

EINVAL

The storage pool ID is not valid.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_igetstoragepool()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached storage pool information was not valid.

GPFS_E_INVALID_ISCAN

The **iscan** parameters were not valid.

Examples

This programming segment gets the storage pool name based on the given storage pool ID. The returned storage pool name is stored in **StoragePoolNameBuffer** which has the length of **StoragePoolNameSize**.

```
gpfs_iscan_t *fsInodeScanP;  
gpfs_igetstoragepool(fsInodeScanP, StgpoolIdBuffer, &StgpoolNameBuffer, StgpoolNameSize);
```

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_iopen() subroutine

Opens a file or directory by inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_ifile_t *gpfs_iopen(gpfs_fssnap_handle_t *fssnapHandle,
                        gpfs_ino_t ino,
                        int open_flags,
                        const gpfs_iattr_t *statxbuf,
                        const char *symLink);
```

Description

The **gpfs_iopen()** subroutine opens a user file or directory for backup. The file is identified by its inode number **ino** within the file system or snapshot identified by the **fssnapHandle**. The **fssnapHandle** parameter must be the same one that was used to create the inode scan that returned the inode number **ino**.

To read the file or directory, the **open_flags** must be set to **GPFS_O_BACKUP**. The **statxbuf** and **symLink** parameters are reserved for future use and must be set to NULL.

For an overview of using **gpfs_iopen()** in a backup application, see “Using APIs to develop backup applications” on page 52.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

ino

The inode number.

open_flags

GPFS_O_BACKUP

Read files for backup.

O_RDONLY

For **gpfs_iread()**.

statxbuf

This parameter is reserved for future use and should always be set to NULL.

symLink

This parameter is reserved for future use and should always be set to NULL.

Exit status

If the **gpfs_iopen()** subroutine is successful, it returns a pointer to the inode's file handle.

gpfs_iopen()

If the **gpfs_iopen()** subroutine is unsuccessful, it returns NULL and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

Missing or incorrect parameter.

| **ENOENT**

| The file does not exist in the file system.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_iopen()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

| **GPFS_E_INVALID_INUM**

| Users are not authorized to open the reserved inodes.

Examples

For an example using **gpfs_iopen()**, see `/usr/lpp/mmfs/samples/util/tsreaddir.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_iopen64() subroutine

Opens a file or directory by inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_ifile_t *gpfs_iopen64(gpfs_fssnap_handle_t *fssnapHandle,
                           gpfs_ino64_t ino,
                           int open_flags,
                           const gpfs_iattr64_t *statxbuf,
                           const char *symLink);
```

Description

The **gpfs_iopen64()** subroutine opens a user file or directory for backup. The file is identified by its inode number **ino** within the file system or snapshot identified by the **fssnapHandle**. The **fssnapHandle** parameter must be the same one that was used to create the inode scan that returned the inode number **ino**.

To read the file or directory, the **open_flags** must be set to **GPFS_O_BACKUP**. The **statxbuf** and **symLink** parameters are reserved for future use and must be set to NULL.

For an overview of using **gpfs_iopen64()** in a backup application, see “Using APIs to develop backup applications” on page 52.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

The file system snapshot handle.

ino

The inode number.

open_flags

GPFS_O_BACKUP

Read files for backup.

O_RDONLY

For **gpfs_iread()**.

statxbuf

This parameter is reserved for future use and should always be set to NULL.

symLink

This parameter is reserved for future use and should always be set to NULL.

Exit status

If the **gpfs_iopen64()** subroutine is successful, it returns a pointer to the inode's file handle.

gpfs_iopen64()

If the `gpfs_iopen64()` subroutine is unsuccessful, it returns NULL and the global error variable `errno` is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EFORMAT

The file system version number is not valid.

EINVAL

Missing or incorrect parameter.

ENOENT

The file does not exist in the file system.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The `gpfs_iopen64()` subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

GPFS_E_INVALID_IATTR

The `iattr` structure was corrupted.

GPFS_E_INVALID_INUM

Users are not authorized to open the reserved inodes.

Note: `gpfs_iopen64()` calls the standard library subroutines `dup()`, `open()`, and `malloc()`; if one of these called subroutines returns an error, `gpfs_iopen64()` also returns that error.

Examples

See the `gpfs_iopen()` example in `/usr/lpp/mmfs/samples/util/tsreaddir.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_iputattrsx() subroutine

Sets the extended file attributes for a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_iputattrsx(gpfs_ifile_t *ifile,
                   int flags,
                   void *buffer,
                   const char *pathName);
```

Description

The `gpfs_iputattrsx()` subroutine, together with `gpfs_igetattrsx()`, is intended for use by a backup program to save (`gpfs_igetattrsx()`) and restore (`gpfs_iputattrsx()`) all of the extended attributes of a file. This subroutine also sets the storage pool for the file and sets data replication to the values that are saved in the extended attributes.

This subroutine can optionally invoke the policy engine to match a **RESTORE** rule using the file's attributes saved in the extended attributes to set the file's storage pool and data replication as when calling `gpfs_fputattrswithpathname()`. When used with the policy engine, the caller should include the full path to the file, including the file name, to allow rule selection based on file name or path.

By default, the routine will not use **RESTORE** policy rules for data placement. The `pathName` parameter will be ignored and may be set to `NULL`.

If the call does not use **RESTORE** policy rules, or if the file fails to match a **RESTORE** rule, or if there are not **RESTORE** rules installed, then the storage pool and data replication are selected as when calling `gpfs_fputattrsx()`.

The buffer passed in should contain extended attribute data that was obtained by a previous call to `gpfs_fgetattrsx()`.

Note: This call will restore extended attributes used for the Data Storage Management (XDSM) API (also known as DMAPI) if they are present in the buffer.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

`ifile`

A pointer to `gpfs_ifile_t` from `gpfs_iopen()`.

`flags`

Flags must have one of the following values:

GPFS_ATTRFLAG_NO_PLACEMENT

File attributes are restored, but the storage pool and data replication are unchanged.

GPFS_ATTRFLAG_IGNORE_POOL

File attributes are restored, but the storage pool and data replication are selected by matching the saved attributes to a placement rule instead of restoring the saved storage pool.

gpfs_iputattrsx()

GPFS_ATTRFLAG_USE_POLICY

File attributes are restored, but the storage pool and data replication are selected by matching the saved attributes to a **RESTORE** rule instead of restoring the saved storage pool.

GPFS_ATTRFLAG_USE_POLICY

Uses the restore policy rules to determine the pool ID.

GPFS_ATTRFLAG_INCL_DMAPI

Includes the DMAPI attributes.

GPFS_ATTRFLAG_FINALIZE_ATTRS

Finalizes immutability attributes.

GPFS_ATTRFLAG_SKIP_IMMUTABLE

Skips immutable attributes.

GPFS_ATTRFLAG_INCL_ENCR

Includes encryption attributes.

GPFS_ATTRFLAG_SKIP_CLONE

Skips clone attributes.

GPFS_ATTRFLAG_MODIFY_CLONEPARENT

Allows modification on the clone parent.

buffer

A pointer to the buffer containing the extended attributes for the file.

pathName

A pointer to a file path and file name. NULL is a valid value for **pathName**.

Note: **pathName** is a UTF-8 encoded string. On Windows, applications can convert UTF-16 (Unicode) to UTF-8 using the platform's **WideCharToMultiByte** function.

Exit status

If the **gpfs_iputattrsx()** subroutine is successful, it returns a value of 0.

If the **gpfs_iputattrsx()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

The buffer pointed to by **buffer** does not contain valid attribute data, or invalid flags were provided.

ENOSYS

The **gpfs_iputattrsx()** subroutine is not supported under the current file system format.

EPERM

The caller of the subroutine must have superuser privilege.

ESTALE

The cached *fs* information was not valid.

GPFS_E_INVALID_IFILE

The **ifile** parameters provided were not valid.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_iread()

gpfs_iread() subroutine

Reads a file opened by `gpfs_iopen()`.

Library

GPFS Library (`libgpfs.a` for AIX, `libgpfs.so` for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_iread(gpfs_ifile_t *ifile,
              void *buffer,
              int bufferSize,
              gpfs_off64_t *offset);
```

Description

The `gpfs_iread()` subroutine reads data from the file indicated by the `ifile` parameter returned from `gpfs_iopen()`. This subroutine reads data beginning at parameter `offset` and continuing for `bufferSize` bytes into the buffer specified by `buffer`. If successful, the subroutine returns a value that is the length of the data read, and sets parameter `offset` to the offset of the next byte to be read. A return value of 0 indicates end-of-file.

For an overview of using `gpfs_iread()` in a backup application, see “Using APIs to develop backup applications” on page 52.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- `libgpfs.a` for AIX
- `libgpfs.so` for Linux

Parameters

`ifile`

Pointer to `gpfs_ifile_t` from `gpfs_iopen()`.

`buffer`

Buffer for the data to be read.

`bufferSize`

Size of the buffer (that is, the amount of data to be read).

`offset`

Offset of where within the file to read. If `gpfs_iread()` is successful, `offset` is updated to the next byte after the last one that was read.

Exit status

If the `gpfs_iread()` subroutine is successful, it returns the number of bytes read.

If the `gpfs_iread()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EISDIR

The specified file is a directory.

EINVAL

Missing or incorrect parameter.

ENOSYS

The **gpfs_iread()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_IFILE

Incorrect **ifile** parameter.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_ireaddir()

gpfs_ireaddir() subroutine

Reads the next directory entry.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_ireaddir(gpfs_ifile_t *idir,
                 const gpfs_direntx_t **dirent);
```

Description

The **gpfs_ireaddir()** subroutine returns the next directory entry in a file system. For an overview of using **gpfs_ireaddir()** in a backup application, see “Using APIs to develop backup applications” on page 52.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

idir

Pointer to **gpfs_ifile_t** from **gpfs_iopen()**.

dirent

Pointer to returned pointer to directory entry.

Exit status

If the **gpfs_ireaddir()** subroutine is successful, it returns a value of 0 and sets the **dirent** parameter to point to the returned directory entry. If there are no more GPFS directory entries, **gpfs_ireaddir()** returns a value of 0 and sets the **dirent** parameter to NULL.

If the **gpfs_ireaddir()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_ireaddir()** subroutine is not available.

ENOTDIR

File is not a directory.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_IFILE

Incorrect **ifile** parameter.

Examples

For an example using **gpfs_ireaddir()**, see `/usr/lpp/mmfs/samples/util/tsreaddir.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_ireaddir64()

gpfs_ireaddir64() subroutine

Reads the next directory entry.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_ireaddir64(gpfs_ifile_t *idir,
                   const gpfs_direntx64_t **dirent);
```

Description

The **gpfs_ireaddir64()** subroutine returns the next directory entry in a file system. For an overview of using **gpfs_ireaddir64()** in a backup application, see “Using APIs to develop backup applications” on page 52.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

idir

A pointer to **gpfs_ifile_t** from **gpfs_iopen64()**.

dirent

A pointer to the returned pointer to the directory entry.

Exit status

If the **gpfs_ireaddir64()** subroutine is successful, it returns a value of 0 and sets the **dirent** parameter to point to the returned directory entry. If there are no more GPFS directory entries, **gpfs_ireaddir64()** returns a value of 0 and sets the **dirent** parameter to NULL.

If the **gpfs_ireaddir64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_ireaddir64()** subroutine is not available.

ENOTDIR

File is not a directory.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_IFILE

Incorrect **ifile** parameter.

Examples

See the **gpfs_ireaddir()** example in `/usr/lpp/mmfs/samples/util/tsreaddir.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_ireadlink()

gpfs_ireadlink() subroutine

Reads a symbolic link by inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_ireadlink(gpfs_fssnap_handle_t *fssnapHandle,
                  gpfs_ino_t ino,
                  char *buffer,
                  int bufferSize);
```

Description

The **gpfs_ireadlink()** subroutine reads a symbolic link by inode number. Like **gpfs_iopen()**, it uses the same **fssnapHandle** parameter that was used by the inode scan.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

File system snapshot handle.

ino

inode number of the link file to read.

buffer

Pointer to buffer for the returned link data.

bufferSize

Size of the buffer.

Exit status

If the **gpfs_ireadlink()** subroutine is successful, it returns the number of bytes read.

If the **gpfs_ireadlink()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

Missing or incorrect parameter.

ENOENT

No such file or directory.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_ireadlink()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

On AIX, the buffer is too small to return the symbolic link.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_ireadlink64()

gpfs_ireadlink64() subroutine

Reads a symbolic link by inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_ireadlink64(gpfs_fssnap_handle_t *fssnapHandle,
                    gpfs_ino64_t ino,
                    char *buffer,
                    int bufferSize);
```

Description

The **gpfs_ireadlink64()** subroutine reads a symbolic link by inode number. Like **gpfs_iopen64()**, it uses the same **fssnapHandle** parameter that was used by the inode scan.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

The file system snapshot handle.

ino

The inode number of the link file to read.

buffer

A pointer to buffer for the returned link data.

bufferSize

The size of the buffer.

Exit status

If the **gpfs_ireadlink64()** subroutine is successful, it returns the number of bytes read.

If the **gpfs_ireadlink64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

Missing or incorrect parameter.

ENOENT

No such file or directory.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_ireadlink64()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

On AIX, the buffer is too small to return the symbolic link.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

Note: **gpfs_ireadlink64()** calls the standard library subroutine **readlink()**; if this called subroutine returns an error, **gpfs_ireadlink64()** also returns that error.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_ireadx()

gpfs_ireadx() subroutine

Performs block level incremental read of a file within an incremental inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_off64_t gpfs_ireadx(gpfs_ifile_t *ifile,
                        gpfs_iscan_t *iscan,
                        void *buffer,
                        int bufferSize,
                        gpfs_off64_t *offset,
                        gpfs_off64_t termOffset,
                        int *hole);
```

Description

The **gpfs_ireadx()** subroutine performs a block level incremental read on a file opened by **gpfs_iopen()** within a given incremental scan opened using **gpfs_open_inodescan()**.

For an overview of using **gpfs_ireadx()** in a backup application, see “Using APIs to develop backup applications” on page 52.

The **gpfs_ireadx()** subroutine returns the data that has changed since the **prev_fssnapId** specified for the inode scan. The file is scanned starting at **offset** and terminating at **termOffset**, looking for changed data. Once changed data is located, the **offset** parameter is set to its location, the new data is returned in the **buffer** provided, and the amount of data returned is the subroutine's value.

If the change to the data is that it has been deleted (that is, the file has been truncated), no data is returned, but the **hole** parameter is returned with a value of 1, and the size of the **hole** is returned as the subroutine's value. The returned size of the hole may exceed the **bufferSize** provided. If no changed data was found before reaching the **termOffset** or the end-of-file, then the **gpfs_ireadx()** subroutine return value is 0.

Block level incremental backups are not available on small files (a file size smaller than the file system block size), directories, or if the file has been deleted. The **gpfs_ireadx()** subroutine can still be used, but it returns all of the file's data, operating like the standard **gpfs_iread()** subroutine. However, the **gpfs_ireadx()** subroutine will still identify sparse files and explicitly return information on holes in the files, rather than returning the NULL data.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

ifile

Pointer to **gpfs_ifile_t** returned from **gpfs_iopen()**.

iscan

Pointer to **gpfs_iscan_t** from **gpfs_open_inodescan()**.

buffer

Pointer to buffer for returned data, or NULL to query the next increment to be read.

bufferSize

Size of buffer for returned data.

offset

On input, the offset to start the scan for changes. On output, the offset of the changed data, if any was detected.

termOffset

Read terminates before reading this offset. The caller may specify **ia_size** from the file's **gpfs_iattr_t** or 0 to scan the entire file.

hole

Pointer to a flag returned to indicate a hole in the file. A value of 0 indicates that the **gpfs_ireadx()** subroutine returned data in the **buffer**. A value of 1 indicates that **gpfs_ireadx()** encountered a hole at the returned **offset**.

Exit status

If the **gpfs_ireadx()** subroutine is successful, it returns the number of bytes read and returned in **bufP**, or the size of the hole encountered in the file.

If the **gpfs_ireadx()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The file system stripe ID from the **iscanId** does not match the **ifile**'s.

EINVAL

Missing or incorrect parameter.

EISDIR

The specified file is a directory.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_ireadx()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

The file system snapshot ID from the **iscanId** is more recent than the **ifile**'s.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_IFILE

Incorrect **ifile** parameter.

GPFS_E_INVALID_ISCAN

Incorrect **iscan** parameter.

gpfs_ireadx()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_iscan_t structure

Contains a handle for an inode scan of a GPFS file system or snapshot.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct gpfs_iscan gpfs_iscan_t;
```

Description

The `gpfs_iscan_t` structure contains a handle for an inode scan of a GPFS file system or snapshot.

Members

`gpfs_iscan`

The handle for an inode scan for a GPFS file system or snapshot.

Examples

For an example using `gpfs_iscan_t`, see `/usr/lpp/mmfs/samples/util/tstimes.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_lib_init()

gpfs_lib_init() subroutine

Sets up a GPFS interface for additional calls.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_lib_init(int flags);
```

Description

The **gpfs_lib_init()** subroutine, together with the **gpfs_lib_term()** subroutine, is intended for use by a program that makes repeated calls to a GPFS programming interface. This subroutine sets up the internal structure to speed up additional interface calls.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

flags

Reserved for future use. Must be zero.

Exit status

If the **gpfs_lib_init()** subroutine is successful, it returns a value of 0.

If the **gpfs_lib_init()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

A nonzero value was passed as the **flags** parameter.

ENOSYS

The **gpfs_lib_init()** subroutine is not supported under the current file system format.

Examples

For an example using **gpfs_lib_init()**, see `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_lib_term() subroutine

Cleans up after GPFS interface calls have been completed.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_lib_term(int flags);
```

Description

The **gpfs_lib_term()** subroutine, together with the **gpfs_lib_init()** subroutine, is intended for use by a program that makes repeated calls to a GPFS programming interface. This subroutine cleans up the internal structure previously set up by **gpfs_lib_init()**.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

flags

Reserved for future use. Must be zero.

Exit status

If the **gpfs_lib_term()** subroutine is successful, it returns a value of 0.

If the **gpfs_lib_term()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINTR

The **gpfs_lib_term()** subroutine was interrupted by a signal that was caught. Cleanup was done.

EINVAL

A nonzero value was passed as the **flags** parameter.

Examples

For an example using **gpfs_lib_term()**, see `/usr/lpp/mmfs/samples/util/tsfindinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_next_inode()

gpfs_next_inode() subroutine

Retrieves the next inode from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_next_inode(gpfs_iscan_t *iscan,
                   gpfs_ino_t termIno,
                   const gpfs_iattr_t **iattr);
```

Description

The **gpfs_next_inode()** subroutine obtains the next inode from the specified inode scan and sets the **iattr** pointer to the inode's attributes. The **termIno** parameter can be used to terminate the inode scan before the last inode in the file system or snapshot being scanned. A value of 0 may be provided to indicate the last inode in the file system or snapshot. If there are no more inodes to be returned before the termination inode, the **gpfs_next_inode()** subroutine returns a value of 0 and the inode's attribute pointer is set to NULL.

For an overview of using **gpfs_next_inode()** in a backup application, see "Using APIs to develop backup applications" on page 52.

To generate a full backup, invoke **gpfs_open_inodescan()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_next_inode()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For an incremental backup, only inodes of files that have changed since the specified previous snapshot will be returned. Any operation that changes the file's **mtime** or **ctime** is considered a change and will cause the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

Incremental backups return deleted files, but full backups do not. A deleted file is indicated by the field **ia_nlinks** having a value of 0.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

Pointer to the inode scan handle.

termIno

The inode scan terminates before this inode number. The caller may specify **maxIno** from **gpfs_open_inodescan()** or zero to scan the entire inode file.

iaatr

Pointer to the returned pointer to the inode's **iaatr**.

Exit status

If the **gpfs_next_inode()** subroutine is successful, it returns a value of 0 and a pointer. The pointer points to NULL if there are no more inodes. Otherwise, the pointer points to the returned inode's attributes.

If the **gpfs_next_inode()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_next_inode()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID is not valid.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Examples

For an example using **gpfs_next_inode()**, see `/usr/lpp/mmfs/samples/util/tstimes.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_next_inode64()

gpfs_next_inode64() subroutine

Retrieves the next inode from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_next_inode64(gpfs_iscan_t *iscan,
                    gpfs_ino64_t termIno,
                    const gpfs_iattr64_t **iattr);
```

Description

The **gpfs_next_inode64()** subroutine obtains the next inode from the specified inode scan and sets the **iattr** pointer to the inode's attributes. The **termIno** parameter can be used to stop the inode scan before the last inode in the file system or snapshot being scanned. A value of 0 can be provided to indicate the last inode in the file system or snapshot. If there are no more inodes to be returned before the termination inode, the **gpfs_next_inode64()** subroutine returns a value of 0 and the inode's attribute pointer is set to NULL.

For an overview of using **gpfs_next_inode64()** in a backup application, see "Using APIs to develop backup applications" on page 52.

To generate a full backup, invoke **gpfs_open_inodescan64()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode64()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_next_inode64()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For an incremental backup, only inodes of files that have changed since the specified previous snapshot will be returned. Any operation that changes the file's **mtime** or **ctime** is considered a change and will cause the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

Incremental backups return deleted files, but full backups do not. A deleted file is indicated by the field **ia_nlinks** having a value of 0.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to the inode scan handle.

termIno

The inode scan terminates before this inode number. The caller may specify **maxIno** from **gpfs_open_inodescan64()** or zero to scan the entire inode file.

iattr

A pointer to the returned pointer to the inode's **iattr**.

Exit status

If the **gpfs_next_inode64()** subroutine is successful, it returns a value of 0 and a pointer. The pointer points to NULL if there are no more inodes. Otherwise, the pointer points to the returned inode's attributes.

If the **gpfs_next_inode64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_next_inode64()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID is not valid.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Examples

See the **gpfs_next_inode()** example in `/usr/lpp/mmfs/samples/util/tstimes.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_next_inode_with_xattrs()

gpfs_next_inode_with_xattrs() subroutine

Retrieves the next inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_next_inode_with_xattrs(gpfs_iscan_t *iscan,
                               gpfs_ino_t termIno,
                               const gpfs_iattr_t **iattr,
                               const char **xattrBuf,
                               unsigned int *xattrBufLen);
```

Description

The **gpfs_next_inode_with_xattrs()** subroutine retrieves the next inode and its extended attributes from the inode scan. The set of extended attributes returned are defined when the inode scan was opened. The scan stops before the last inode that was specified or the last inode in the inode file being scanned.

The data returned by **gpfs_next_inode()** is overwritten by subsequent calls to **gpfs_next_inode()**, **gpfs_seek_inode()**, or **gpfs_stat_inode()**.

The **termIno** parameter provides a way to partition an inode scan so it can be run on more than one node.

The returned values for **xattrBuf** and **xattrBufLen** must be provided to **gpfs_next_xattr()** to obtain the extended attribute names and values. The buffer used for the extended attributes is overwritten by subsequent calls to **gpfs_next_inode()**, **gpfs_seek_inode()**, or **gpfs_stat_inode()**.

The returned pointers to the extended attribute name and value will be aligned to a double-word boundary.

Parameters

iscan

A pointer to the inode scan descriptor.

termIno

The inode scan stops before this inode number. The caller can specify **maxIno** from **gpfs_open_inodescan()** or zero to scan the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

xattrBuf

A pointer to the returned pointer to the **xiattr** buffer.

xattrBufLen

The returned length of the **xiattr** buffer.

Exit status

If the **gpfs_next_inode_with_xattrs()** subroutine is successful, it returns a value of 0 and **iattr** is set to point to **gpfs_iattr_t**. The pointer points to NULL if there are no more inodes, otherwise, the pointer points to **gpfs_iattr_t**.

If the **gpfs_next_inode_with_xattrs()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to NULL to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EFAULT

The buffer data was overwritten.

ENOMEM

The buffer is too small, unable to allocate memory for the request.

ENOSYS

The **gpfs_next_inode_with_xattrs()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

GPFS_E_INVALID_XATTR

Incorrect parameters.

Examples

For an example using **gpfs_next_inode_with_xattrs()**, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_next_inode_with_xattrs64()

gpfs_next_inode_with_xattrs64() subroutine

Retrieves the next inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_next_inode_with_xattrs64(gpfs_iscan_t *iscan,
                                gpfs_ino64_t termIno,
                                const gpfs_iattr64_t **iattr,
                                const char **xattrBuf,
                                unsigned int *xattrBufLen);
```

Description

The **gpfs_next_inode_with_xattrs64()** subroutine retrieves the next inode and its extended attributes from the inode scan. The set of extended attributes returned are defined when the inode scan was opened. The scan stops before the last inode that was specified or the last inode in the inode file being scanned.

The data returned by **gpfs_next_inode64()** is overwritten by subsequent calls to **gpfs_next_inode64()**, **gpfs_seek_inode64()**, or **gpfs_stat_inode64()**.

The **termIno** parameter provides a way to partition an inode scan so it can be run on more than one node.

The returned values for **xattrBuf** and **xattrBufLen** must be provided to **gpfs_next_xattr()** to obtain the extended attribute names and values. The buffer used for the extended attributes is overwritten by subsequent calls to **gpfs_next_inode64()**, **gpfs_seek_inode64()**, or **gpfs_stat_inode64()**.

The returned pointers to the extended attribute name and value will be aligned to a double-word boundary.

Parameters

iscan

A pointer to the inode scan descriptor.

termIno

The inode scan stops before this inode number. The caller can specify **maxIno** from **gpfs_open_inodescan64()** or zero to scan the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

xattrBuf

A pointer to the returned pointer to the **xiattr** buffer. Initialize this parameter to a valid value or NULL before calling **gpfs_next_inode_with_xattrs64**.

xattrBufLen

The returned length of the **xiattr** buffer. Initialize this parameter to a valid value or NULL before calling **gpfs_next_inode_with_xattrs64**.

Exit status

If the **gpfs_next_inode_with_xattrs64()** subroutine is successful, it returns a value of 0 and **iattr** is set to point to **gpfs_iattr_t**. The pointer points to NULL if there are no more inodes, otherwise, the pointer points to **gpfs_iattr_t**.

If the **gpfs_next_inode_with_xattrs64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to NULL to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EFAULT

The buffer data was overwritten.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_next_inode_with_xattrs64()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

GPFS_E_INVALID_XATTR

Incorrect parameters.

Examples

See the **gpfs_next_inode_with_xattrs()** example in `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_next_xattr()

gpfs_next_xattr() subroutine

Returns individual attributes and their values.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_next_xattr(gpfs_iscan_t *iscan,
                  const char **xattrBuf,
                  unsigned int *xattrBufLen,
                  const char **name,
                  unsigned int *valueLen,
                  const char **value);
```

Description

The **gpfs_next_xattr()** subroutine iterates over the extended attributes buffer returned by the **gpfs_next_inode_with_xattrs()** or **gpfs_next_inode_with_xattrs64()** subroutine to return the individual attributes and their values. The attribute names are null-terminated strings, whereas the attribute value contains binary data.

Note: The caller is not allowed to modify the returned attribute names or values. The data returned by **gpfs_next_xattr()** might be overwritten by subsequent calls to **gpfs_next_xattr()** or other GPFS library calls.

Parameters

iscan

A pointer to the inode descriptor.

xattrBuf

A pointer to the pointer to the attribute buffer.

xattrBufLen

A pointer to the attribute buffer length.

name

A pointer to the attribute name.

valueLen

A pointer to the length of the attribute value.

value

A pointer to the attribute value.

Exit status

If the **gpfs_next_xattr()** subroutine is successful, it returns a value of 0 and a pointer to the attribute name. It also sets:

- The **valueLen** parameter to the length of the attribute value
- The **value** parameter to point to the attribute value
- The **xattrBufLen** parameter to the remaining length of buffer
- The **xattrBuf** parameter to index the next attribute in buffer

If the **gpfs_next_xattr()** subroutine is successful, but there are no more attributes in the buffer, it returns a value of 0 and the attribute name is set to NULL. It also sets:

- The **valueLen** parameter to 0
- The **value** parameter to NULL
- The **xattrBufLen** parameter to 0
- The **xattrBuf** parameter to NULL

If the **gpfs_next_xattr()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

Incorrect parameters.

ENOSYS

The **gpfs_next_xattr()** subroutine is not available.

Examples

For an example using **gpfs_next_xattr()**, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_opaque_acl_t structure

Contains buffer mapping for the `gpfs_getacl()` and `gpfs_putacl()` subroutines.

Library

GPFS Library (`libgpfs.a` for AIX, `libgpfs.so` for Linux)

Structure

```
typedef struct
{
    int          acl_buffer_len;
    unsigned short acl_version;
    unsigned char acl_type;
    char         acl_var_data[1];
} gpfs_opaque_acl_t;
```

Description

The `gpfs_opaque_acl_t` structure contains size, version, and ACL type information for the `gpfs_getacl()` and `gpfs_putacl()` subroutines.

Members

`acl_buffer_len`

On input, this field must be set to the total length, in bytes, of the data structure being passed to GPFS. On output, this field contains the actual size of the requested information. If the initial size of the buffer is not large enough to contain all of the information, the `gpfs_getacl()` invocation must be repeated with a larger buffer.

`acl_version`

This field contains the current version of the GPFS internal representation of the ACL. On input to the `gpfs_getacl()` subroutine, set this field to zero.

`acl_type`

On input to the `gpfs_getacl()` subroutine, set this field to either `GPFS_ACL_TYPE_ACCESS` or `GPFS_ACL_TYPE_DEFAULT`, depending on which ACL is requested. These constants are defined in the `gpfs.h` header file.

`acl_var_data`

This field signifies the beginning of the remainder of the ACL information.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_open_inodescan() subroutine

Opens an inode scan of a file system or snapshot.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_iscan_t *gpfs_open_inodescan(gpfs_fssnap_handle_t *fssnapHandle,
                                 const gpfs_fssnap_id_t *prev_fssnapId,
                                 gpfs_ino_t *maxIno);
```

Description

The **gpfs_open_inodescan()** subroutine opens a scan of the inodes in the file system or snapshot identified by the **fssnapHandle** parameter. The scan traverses all user files, directories and links in the file system or snapshot. The scan begins with the user file with the lowest inode number and returns the files in increasing order. The **gpfs_seek_inode()** subroutine may be used to set the scan position to an arbitrary inode. System files, such as the block allocation maps, are omitted from the scan. The file system must be mounted to open an inode scan.

For an overview of using **gpfs_open_inodescan()** in a backup application, see “Using APIs to develop backup applications” on page 52.

To generate a full backup, invoke **gpfs_open_inodescan()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_open_inodescan()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For the incremental backup, any operation that changes the file's **mtime** or **ctime** causes the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

A full inode scan (**prev_fssnapId** set to NULL) does not return any inodes of nonexistent or deleted files, but an incremental inode scan (**prev_fssnapId** not NULL) does return inodes for files that have been deleted since the previous snapshot. The inodes of deleted files have a link count of zero.

If the snapshot indicated by **prev_fssnapId** is available, the caller may benefit from the extended read subroutine, **gpfs_ireadx()**, which returns only the changed blocks within the files. Without the previous snapshot, all blocks within the changed files are returned.

Once a full or incremental backup completes, the **new_fssnapId** must be saved in order to reuse it on a subsequent incremental backup. This **fssnapId** must be provided to the **gpfs_open_inodescan()** subroutine, as the **prev_fssnapId** input parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

gpfs_open_inodescan()

Parameters

fssnapHandle

File system snapshot handle.

prev_fssnapId

Pointer to file system snapshot ID or NULL. If **prev_fssnapId** is provided, the inode scan returns only the files that have changed since the previous backup. If the pointer is NULL, the inode scan returns all user files.

maxIno

Pointer to inode number or NULL. If provided, **gpfs_open_inodescan()** returns the maximum inode number in the file system or snapshot being scanned.

Exit status

If the **gpfs_open_inodescan()** subroutine is successful, it returns a pointer to an inode scan handle.

If the **gpfs_open_inodescan()** subroutine is unsuccessful, it returns a NULL pointer and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The file system snapshot ID passed for **prev_fssnapId** is from a different file system.

EINVAL

Incorrect parameters.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_open_inodescan()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

The **prev_fssnapId** parameter is the same as or more recent than **snapId** being scanned.

ESTALE

Cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID passed for **prev_fssnapId** is not valid.

Examples

For an example using **gpfs_open_inodescan()**, see `/usr/lpp/mmfs/samples/util/tstimes.c`.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_open_inodescan64()

gpfs_open_inodescan64() subroutine

Opens an inode scan of a file system or snapshot.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_iscan_t *gpfs_open_inodescan64(gpfs_fssnap_handle_t *fssnapHandle,
                                   const gpfs_fssnap_id_t *prev_fssnapId,
                                   gpfs_ino64_t *maxIno);
```

Description

The **gpfs_open_inodescan64()** subroutine opens a scan of the inodes in the file system or snapshot identified by the **fssnapHandle** parameter. The scan traverses all user files, directories and links in the file system or snapshot. The scan begins with the user file with the lowest inode number and returns the files in increasing order. The **gpfs_seek_inode64()** subroutine may be used to set the scan position to an arbitrary inode. System files, such as the block allocation maps, are omitted from the scan. The file system must be mounted to open an inode scan.

For an overview of using **gpfs_open_inodescan64()** in a backup application, see “Using APIs to develop backup applications” on page 52.

To generate a full backup, invoke **gpfs_open_inodescan64()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode64()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_open_inodescan64()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For the incremental backup, any operation that changes the file's **mtime** or **ctime** causes the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

A full inode scan (**prev_fssnapId** set to NULL) does not return any inodes of nonexistent or deleted files, but an incremental inode scan (**prev_fssnapId** not NULL) does return inodes for files that have been deleted since the previous snapshot. The inodes of deleted files have a link count of zero.

If the snapshot indicated by **prev_fssnapId** is available, the caller may benefit from the extended read subroutine, **gpfs_ireadx()**, which returns only the changed blocks within the files. Without the previous snapshot, all blocks within the changed files are returned.

Once a full or incremental backup completes, the **new_fssnapId** must be saved in order to reuse it on a subsequent incremental backup. This **fssnapId** must be provided to the **gpfs_open_inodescan64()** subroutine, as the **prev_fssnapId** input parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fssnapHandle

The file system snapshot handle.

prev_fssnapId

A pointer to file system snapshot ID or NULL. If **prev_fssnapId** is provided, the inode scan returns only the files that have changed since the previous backup. If the pointer is NULL, the inode scan returns all user files.

maxIno

A pointer to inode number or NULL. If provided, **gpfs_open_inodescan64()** returns the maximum inode number in the file system or snapshot being scanned.

Exit status

If the **gpfs_open_inodescan64()** subroutine is successful, it returns a pointer to an inode scan handle.

If the **gpfs_open_inodescan64()** subroutine is unsuccessful, it returns a NULL pointer and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The file system snapshot ID passed for **prev_fssnapId** is from a different file system.

EINVAL

Incorrect parameters.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_open_inodescan64()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

The **prev_fssnapId** parameter is the same as or more recent than **snapId** being scanned.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID passed for **prev_fssnapId** is not valid.

Note: **gpfs_open_inodescan64()** calls the standard library subroutines **dup()** and **malloc()**; if one of these called subroutines returns an error, **gpfs_open_inodescan64()** also returns that error.

Examples

See the **gpfs_open_inodescan()** example in `/usr/lpp/mmfs/samples/util/tstimes.c`.

gpfs_open_inodescan64()

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_open_inodescan_with_xattrs() subroutine

Opens an inode file and extended attributes for an inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_iscan_t *gpfs_open_inodescan_with_xattrs(gpfs_fssnap_handle_t *fssnapHandle,
                                             const gpfs_fssnap_id_t *prev_fssnapId,
                                             int nxAttrs,
                                             const char *xattrsList[],
                                             gpfs_ino_t *maxIno);
```

Description

The **gpfs_open_inodescan_with_xattrs()** subroutine opens an inode file and extended attributes for an inode scan identified by the **fssnapHandle** parameter. The scan traverses all user files, directories and links in the file system or snapshot. The scan begins with the user file with the lowest inode number and returns the files in increasing order. The **gpfs_seek_inode()** subroutine can be used to set the scan position to an arbitrary inode. System files, such as the block allocation maps, are omitted from the scan. The file system must be mounted to open an inode scan.

For an overview of using **gpfs_open_inodescan_with_xattrs()** in a backup application, see “Using APIs to develop backup applications” on page 52.

To generate a full backup, invoke **gpfs_open_inodescan_with_xattrs()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_open_inodescan_with_xattrs()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For the incremental backup, any operation that changes the file's **mtime** or **ctime** causes the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

A full inode scan (**prev_fssnapId** set to NULL) returns all inodes of existing files. An incremental inode scan (**prev_fssnapId** not NULL) returns inodes for files that have changed since the previous snapshot. The inodes of deleted files have a link count of zero.

If the snapshot indicated by **prev_fssnapId** is available, the caller may benefit from the extended read subroutine, **gpfs_ireadx()**, which returns only the changed blocks within the files. Without the previous snapshot, all blocks within the changed files are returned.

Once a full or incremental backup completes, the **new_fssnapId** must be saved in order to reuse it on a subsequent incremental backup. This **fssnapId** must be provided to the **gpfs_open_inodescan_with_xattrs()** subroutine, as the **prev_fssnapId** input parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX

gpfs_open_inodescan_with_xattrs()

- libgpfs.so for Linux

Parameters

fssnapHandle

The file system snapshot handle.

prev_fssnapId

A pointer to file system snapshot ID or NULL. If **prev_fssnapId** is provided, the inode scan returns only the files that have changed since the previous backup. If the pointer is NULL, the inode scan returns all user files.

nxAttrs

The count of extended attributes to be returned. If **nxAttrs** is set to 0, call returns no extended attributes, like **gpfs_open_inodescan()**. If **nxAttrs** is set to -1, call returns all extended attributes.

xattrsList

A pointer to an array of pointers to names of extended attributes to be returned. **nxAttrsList** may be null if **nxAttrs** is set to 0 or -1.

maxIno

A pointer to inode number or NULL. If provided, **gpfs_open_inodescan_with_xattrs()** returns the maximum inode number in the file system or snapshot being scanned.

Exit status

If the **gpfs_open_inodescan_with_xattrs()** subroutine is successful, it returns a pointer to **gpfs_iscan_t**.

If the **gpfs_open_inodescan_with_xattrs()** subroutine is unsuccessful, it returns a NULL pointer and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The file system snapshot ID passed for **prev_fssnapId** is from a different file system.

EINVAL

Incorrect parameters.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_open_inodescan_with_xattrs()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

The **prev_fssnapId** parameter is the same as or more recent than **snapId** being scanned.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID passed for `prev_fssnapId` is not valid.

Note: `gpfs_open_inodescan_with_xattrs()` calls the standard library subroutines `dup()` and `malloc()`; if one of these called subroutines returns an error, `gpfs_open_inodescan_with_xattrs()` also returns that error.

Examples

For an example using `gpfs_open_inodescan_with_xattrs()`, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_open_inodescan_with_xattrs64()

gpfs_open_inodescan_with_xattrs64() subroutine

Opens an inode file and extended attributes for an inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
gpfs_iscan_t *gpfs_open_inodescan_with_xattrs64(gpfs_fssnap_handle_t *fssnapHandle,
                                               const gpfs_fssnap_id_t *prev_fssnapId,
                                               int nxAttrs,
                                               const char *xattrList[],
                                               gpfs_ino64_t *maxIno);
```

Description

The **gpfs_open_inodescan_with_xattrs64()** subroutine opens an inode file and extended attributes for an inode scan identified by the **fssnapHandle** parameter. The scan traverses all user files, directories and links in the file system or snapshot. The scan begins with the user file with the lowest inode number and returns the files in increasing order. The **gpfs_seek_inode64()** subroutine may be used to set the scan position to an arbitrary inode. System files, such as the block allocation maps, are omitted from the scan. The file system must be mounted to open an inode scan.

For an overview of using **gpfs_open_inodescan_with_xattrs64()** in a backup application, see “Using APIs to develop backup applications” on page 52.

To generate a full backup, invoke **gpfs_open_inodescan_with_xattrs64()** with NULL for the **prev_fssnapId** parameter. Repeated invocations of **gpfs_next_inode64()** then return inode information about all existing user files, directories, and links in inode number order.

To generate an incremental backup, invoke **gpfs_open_inodescan_with_xattrs64()** with the **fssnapId** that was obtained from a **fssnapHandle** at the time the previous backup was created. The snapshot that was used for the previous backup does not need to exist at the time the incremental backup is generated. That is, the backup application needs to remember only the **fssnapId** of the previous backup; the snapshot itself can be deleted as soon as the backup is completed.

For the incremental backup, any operation that changes the file's **mtime** or **ctime** causes the file to be included. Files with no changes to the file's data or file attributes, other than a change to **atime**, are omitted from the scan.

A full inode scan (**prev_fssnapId** set to NULL) returns all inodes of existing files. An incremental inode scan (**prev_fssnapId** not NULL) returns inodes for files that have changed since the previous snapshot. The inodes of deleted files have a link count of zero.

If the snapshot indicated by **prev_fssnapId** is available, the caller may benefit from the extended read subroutine, **gpfs_ireadx()**, which returns only the changed blocks within the files. Without the previous snapshot, all blocks within the changed files are returned.

Once a full or incremental backup completes, the **new_fssnapId** must be saved in order to reuse it on a subsequent incremental backup. This **fssnapId** must be provided to the **gpfs_open_inodescan_with_xattrs64()** subroutine, as the **prev_fssnapId** input parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX

- libgpfs.so for Linux

Parameters

fssnapHandle

The file system snapshot handle.

prev_fssnapId

A pointer to file system snapshot ID or NULL. If **prev_fssnapId** is provided, the inode scan returns only the files that have changed since the previous backup. If the pointer is NULL, the inode scan returns all user files.

nxAttrs

The count of extended attributes to be returned. If **nxAttrs** is set to 0, call returns no extended attributes, like **gpfs_open_inodescan64()**. If **nxAttrs** is set to -1, call returns all extended attributes

xattrsList

A pointer to an array of pointers to names of extended attributes to be returned. **nxAttrsList** may be null if **nxAttrs** is set to 0 or -1.

maxIno

A pointer to inode number or NULL. If provided, **gpfs_open_inodescan_with_xattrs64()** returns the maximum inode number in the file system or snapshot being scanned.

Exit status

If the **gpfs_open_inodescan_with_xattrs64()** subroutine is successful, it returns a pointer to **gpfs_iscan_t**.

If the **gpfs_open_inodescan_with_xattrs64()** subroutine is unsuccessful, it returns a NULL pointer and the global error variable **errno** is set to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EDOM

The file system snapshot ID passed for **prev_fssnapId** is from a different file system.

EINVAL

Incorrect parameters.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_open_inodescan_with_xattrs64()** subroutine is not available.

EPERM

The caller does not have superuser privileges.

ERANGE

The **prev_fssnapId** parameter is the same as or more recent than **snapId** being scanned.

ESTALE

The cached file system information was not valid.

GPFS_E_INVALID_FSSNAPHANDLE

The file system snapshot handle is not valid.

gpfs_open_inodescan_with_xattrs64()

GPFS_E_INVALID_FSSNAPID

The file system snapshot ID passed for `prev_fssnapId` is not valid.

Note: `gpfs_open_inodescan_with_xattrs64()` calls the standard library subroutines `dup()` and `malloc()`; if one of these called subroutines returns an error, `gpfs_open_inodescan_with_xattrs64()` also returns that error.

Examples

See the `gpfs_open_inodescan_with_xattrs()` example in `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_prealloc() subroutine

Pre-allocates disk storage for a GPFS file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_prealloc(gpfs_file_t fileDesc,
                 gpfs_off64_t startOffset,
                 gpfs_off64_t bytesToPrealloc);
```

Description

The **gpfs_prealloc()** subroutine is used to preallocate disk storage for a file that has already been opened, prior to writing data to the file. The preallocated disk storage is started at the requested offset, **startOffset**, and covers at least the number of bytes requested, **bytesToPrealloc**. Allocations are rounded to GPFS block boundaries.

Pre-allocating disk space for a file provides an efficient method for allocating storage without having to write any data. This can result in faster I/O compared to a file which gains disk space incrementally as it grows.

Existing data in the file is not modified. Reading any of the preallocated blocks returns zeroes.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

fileDesc

An integer specifying the file descriptor returned by **open()**.

The file designated for preallocation must be opened for writing.

startOffset

The byte offset into the file at which to begin preallocation.

bytesToPrealloc

The number of bytes to be preallocated.

Exit status

If the **gpfs_prealloc()** subroutine is successful, it returns a value of 0.

If the **gpfs_prealloc()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error. If **errno** is set to one of the following, some storage may have been preallocated:

- EDQUOT
- ENOSPC

The only way to tell how much space was actually preallocated is to invoke the **stat()** subroutine and compare the reported file size and number of blocks used with their values prior to preallocation.

gpfs_prealloc()

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCES

The file is not opened for writing.

EBADF

The file descriptor is not valid.

EDQUOT

A disk quota has been exceeded

EINVAL

The file descriptor does not refer to a GPFS file or a regular file; a negative value was specified for **startOffset** or **bytesToPrealloc**.

ENOSPC

The file system has run out of disk space.

ENOSYS

The **gpfs_prealloc()** subroutine is not supported under the current file system format.

Examples

```
#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <gpfs.h>

int rc;
int fileHandle = -1;
char* fileNameP = "datafile";
offset_t startOffset = 0;
offset_t bytesToAllocate = 20*1024*1024; /* 20 MB */

fileHandle = open(fileNameP, O_RDWR|O_CREAT, 0644);
if (fileHandle < 0)
{
    perror(fileNameP);
    exit(1);
}

rc = gpfs_prealloc(fileHandle, startOffset, bytesToAllocate);
if (rc < 0)
{
    fprintf(stderr, "Error %d preallocation at %lld for %lld in %s\n",
        errno, startOffset, bytesToAllocate, fileNameP);
    exit(1);
}
```

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_putacl() subroutine

Restores the access control information for a GPFS file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_putacl(const char *pathname,
               int flags,
               void *acl);
```

Description

The **gpfs_putacl()** subroutine together with the **gpfs_getacl()** subroutine is intended for use by a backup program to save (**gpfs_getacl()**) and restore (**gpfs_putacl()**) the ACL information for the file.

Notes:

1. The use of **gpfs_fgetattns()** and **gpfs_fputattns()** is preferred.
2. You must have **write** access to the file.
3. Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:
 - libgpfs.a for AIX
 - libgpfs.so for Linux

Parameters

pathname

Path name of the file for which the ACLs is to be set.

flags

Consists of one of these values:

- 0** Indicates that the **acl** parameter is to be mapped with the **gpfs_opaque_acl_t** structure. The **gpfs_opaque_acl_t** structure should be used by backup and restore programs.

GPFS_PUTACL_STRUCT

Indicates that the **acl** parameter is to be mapped with the **gpfs_acl_t** structure.

The **gpfs_acl_t** structure is provided for applications that need to change the ACL.

acl

Pointer to a buffer mapped by the structure **gpfs_opaque_acl_t** or **gpfs_acl_t**, depending on the value of **flags**.

This is where the ACL data is stored, and should be the result of a previous invocation of **gpfs_getacl()**.

Exit status

If the **gpfs_putacl()** subroutine is successful, it returns a value of 0.

If the **gpfs_putacl()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

gpfs_putacl()

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

The path name does not refer to a GPFS file or a regular file.

ENOMEM

Unable to allocate memory for the request.

ENOSYS

The **gpfs_putacl()** subroutine is not supported under the current file system format.

ENOTDIR

File is not a directory.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_quotactl() subroutine

Manipulates disk quotas on file systems.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_quotactl(const char *pathname,
                 int cmd,
                 int id,
                 void *bufferP);
```

Description

The **gpfs_quotactl()** subroutine manipulates disk quotas. It enables, disables, and manipulates disk quotas for file systems on which quotas have been enabled.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

pathname

Specifies the path name of any file within the mounted file system to which the quota control command is to applied.

cmd

Specifies the quota control command to be applied and whether it is applied to a user, group, or fileset quota.

The **cmd** parameter can be constructed using **GPFS_QCMD(qcmd, Type)** contained in **gpfs.h**. The **qcmd** parameter specifies the quota control command. The **Type** parameter specifies one of the following quota types:

- user (**GPFS_USRQUOTA**)
- group (**GPFS_GRPQUOTA**)
- fileset (**GPFS_FILESETQUOTA**)

The valid values for the **qcmd** parameter specified in **gpfs.h** are:

Q_QUOTAON

Enables quotas.

Enables disk quotas for the file system specified by the **pathname** parameter and type specified in **Type**. The **id** and **bufferP** parameters are unused. Root user authority is required to enable quotas.

Q_QUOTAOFF

Disables quotas.

Disables disk quotas for the file system specified by the **pathname** parameter and type specified in **Type**. The **id** and **bufferP** parameters are unused. Root user authority is required to disable quotas.

Q_GETQUOTA

Gets quota limits and usage information.

gpfs_quotactl()

Retrieves quota limits and current usage for a user, group, or fileset specified by the `id` parameter. The `bufferP` parameter points to a `gpfs_quotaInfo_t` structure to hold the returned information. The `gpfs_quotaInfo_t` structure is defined in `gpfs.h`.

Root authority is required if the `id` value is not the current id (user id for `GPFS_USRQUOTA`, group id for `GPFS_GRPQUOTA`) of the caller.

Q_SETQUOTA

Sets quota limits

Sets disk quota limits for a user, group, or fileset specified by the `id` parameter. The `bufferP` parameter points to a `gpfs_quotaInfo_t` structure containing the new quota limits. The `gpfs_quotaInfo_t` structure is defined in `gpfs.h`. Root user authority is required to set quota limits.

Q_SETUSE

Sets quota usage

Sets disk quota usage for a user, group, or fileset specified by the `id` parameter. The `bufferP` parameter points to a `gpfs_quotaInfo_t` structure containing the new quota usage. The `gpfs_quotaInfo_t` structure is defined in `gpfs.h`. Root user authority is required to set quota usage.

Q_SYNC

Synchronizes the disk copy of a file system quota

Updates the on disk copy of quota usage information for a file system. The `id` and `bufferP` parameters are unused. Root user authority is required to synchronize a file system quota.

`id` Specifies the user, group, or fileset ID to which the quota control command applies. The `id` parameter is interpreted by the specified quota type.

bufferP

Points to the address of an optional, command-specific data structure that is copied in or out of the system.

Exit status

If the `gpfs_quotactl()` subroutine is successful, it returns a value of 0.

If the `gpfs_quotactl()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EACCES

Search permission is denied for a component of a path prefix.

EFAULT

An invalid `bufferP` parameter is supplied. The associated structure could not be copied in or out of the kernel.

EINVAL

One of the following errors:

- The file system is not mounted.

- Invalid command or quota type.
- Invalid input limits: negative limits or soft limits are greater than hard limits.
- UID is not defined.

ENOENT

No such file or directory.

EPERM

The quota control command is privileged and the caller did not have root user authority.

GPFS_E_NO_QUOTA_INST

The file system does not support quotas. This is the actual **errno** generated by GPFS.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_quotaInfo_t structure

Contains buffer mapping for the `gpfs_quotactl()` subroutine.

Library

GPFS Library (`libgpfs.a` for AIX, `libgpfs.so` for Linux)

Structure

```
typedef struct gpfs_quotaInfo
{
    gpfs_off64_t blockUsage;      /* current block count */
    gpfs_off64_t blockHardLimit; /* absolute limit on disk blks alloc */
    gpfs_off64_t blockSoftLimit; /* preferred limit on disk blks */
    gpfs_off64_t blockInDoubt;   /* distributed shares + "lost" usage for blks */
    int          inodeUsage;     /* current # allocated inodes */
    int          inodeHardLimit; /* absolute limit on allocated inodes */
    int          inodeSoftLimit; /* preferred inode limit */
    int          inodeInDoubt;   /* distributed shares + "lost" usage for inodes */
    gpfs_uid_t  quoId;          /* uid, gid or fileset id */
    int         entryType;      /* entry type, not used */
    unsigned int blockGraceTime; /* time limit for excessive disk use */
    unsigned int inodeGraceTime; /* time limit for excessive inode use */
} gpfs_quotaInfo_t;
```

Description

The `gpfs_quotaInfo_t` structure contains detailed information for the `gpfs_quotactl()` subroutine.

Members

blockUsage

The current block count in 1 KB units.

blockHardLimit

The absolute limit on disk block allocation.

blockSoftLimit

The preferred limit on disk block allocation.

blockInDoubt

The distributed shares and block usage that have not been not accounted for.

inodeUsage

The current number of allocated inodes.

inodeHardLimit

The absolute limit on allocated inodes.

inodeSoftLimit

The preferred inode limit.

inodeInDoubt

The distributed inode share and inode usage that have not been accounted for.

quoId

The user ID, group ID, or fileset ID.

entryType

Not used

blockGraceTime

The time limit (in seconds since the Epoch) for excessive disk use.

inodeGraceTime

The time limit (in seconds since the Epoch) for excessive inode use.

Epoch is midnight on January 1, 1970 UTC (Coordinated Universal Time).

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_seek_inode()

gpfs_seek_inode() subroutine

Advances an inode scan to the specified inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_seek_inode(gpfs_iscan_t *iscan,
                   gpfs_ino_t ino);
```

Description

The `gpfs_seek_inode()` subroutine advances an inode scan to the specified inode number.

The `gpfs_seek_inode()` subroutine is used to start an inode scan at some place other than the beginning of the inode file. This is useful to restart a partially completed backup or an interrupted dump transfer to a mirror. It could also be used to do an inode scan in parallel from multiple nodes, by partitioning the inode number space into separate ranges for each participating node. The maximum inode number is returned when the scan was opened and each invocation to obtain the next inode specifies a termination inode number to avoid returning the same inode more than once.

Note: Compile any program that uses this subroutine with the `-lgpfs` flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

Pointer to the inode scan handle.

ino

The next inode number to be scanned.

Exit status

If the `gpfs_seek_inode()` subroutine is successful, it returns a value of 0.

If the `gpfs_seek_inode()` subroutine is unsuccessful, it returns a value of -1 and sets the global error variable `errno` to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The `gpfs_seek_inode()` subroutine is not available.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Examples

For an example using `gpfs_seek_inode()`, see `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_seek_inode64()

gpfs_seek_inode64() subroutine

Advances an inode scan to the specified inode number.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_seek_inode64(gpfs_iscan_t *iscan,
                    gpfs_ino64_t ino);
```

Description

The **gpfs_seek_inode64()** subroutine advances an inode scan to the specified inode number.

The **gpfs_seek_inode64()** subroutine is used to start an inode scan at some place other than the beginning of the inode file. This is useful to restart a partially completed backup or an interrupted dump transfer to a mirror. It could also be used to do an inode scan in parallel from multiple nodes, by partitioning the inode number space into separate ranges for each participating node. The maximum inode number is returned when the scan was opened and each invocation to obtain the next inode specifies a termination inode number to avoid returning the same inode more than once.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to the inode scan handle.

ino

The next inode number to be scanned.

Exit status

If the **gpfs_seek_inode64()** subroutine is successful, it returns a value of 0.

If the **gpfs_seek_inode64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS

The **gpfs_seek_inode64()** subroutine is not available.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Examples

See the `gpfs_seek_inode()` example in `/usr/lpp/mmfs/samples/util/tsinode.c`.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfs_stat()

gpfs_stat() subroutine

Returns exact file status for a GPFS file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_stat(const char *pathname,
              gpfs_stat64_t *buffer);
```

Description

The **gpfs_stat()** subroutine is used to obtain exact information about the file named by the **pathname** parameter. This subroutine is provided as an alternative to the **stat()** subroutine, which may not provide exact **mtime** and **atime** values. See “Exceptions to Open Group technical standards” on page 881.

read, **write**, or **execute** permission for the named file is not required, but all directories listed in the path leading to the file must be searchable. The file information is written to the area specified by the **buffer** parameter.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

pathname

The path identifying the file for which exact status information is requested.

buffer

A pointer to the **gpfs_stat64_t** structure in which the information is returned. The **gpfs_stat64_t** structure is described in the `sys/stat.h` file.

Exit status

If the **gpfs_stat()** subroutine is successful, it returns a value of 0.

If the **gpfs_stat()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EINVAL

The path name does not refer to a GPFS file or a regular file.

ENOENT

The file does not exist.

ENOSYS

The **gpfs_stat()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_stat_inode()

gpfs_stat_inode() subroutine

Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_stat_inode(gpfs_iscan_t *iscan,
                  gpfs_ino_t ino,
                  gpfs_ino_t termIno,
                  const gpfs_iattr_t **iattr);
```

Description

The **gpfs_stat_inode()** subroutine is used to seek the specified inode and to retrieve that inode and its extended attributes from the inode scan. This subroutine combines **gpfs_seek_inode()** and **get_next_inode()**, but will only return the specified inode.

The data returned by **gpfs_next_inode()** is overwritten by subsequent calls to **gpfs_next_inode()**, **gpfs_seek_inode()**, or **gpfs_stat_inode()**.

The **termIno** parameter provides a way to partition an inode scan so it can be run on more than one node. It is only used by this call to control prefetching.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to an inode scan descriptor.

ino

The inode number to be returned.

termIno

Prefetches inodes up to this inode. The caller might specify **maxIno** from **gpfs_open_inodescan()** or 0 to allow prefetching over the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

Exit status

If the **gpfs_stat_inode()** subroutine is successful, it returns a value of 0 and the **iattr** parameter is set to point to **gpfs_iattr_t**. If the **gpfs_stat_inode()** subroutine is successful, but there are no more inodes before the **termIno** parameter, or if the requested inode does not exist, it returns a value of 0 and the **iattr** parameter is set to NULL.

If the **gpfs_stat_inode()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EPERM

The caller must have superuser privilege.

ENOSYS

The **gpfs_stat_inode()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

ENOMEM

The buffer is too small.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_stat_inode64()

gpfs_stat_inode64() subroutine

Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_stat_inode64(gpfs_iscan_t *iscan,
                    gpfs_ino64_t ino,
                    gpfs_ino64_t termIno,
                    const gpfs_iattr64_t **iattr);
```

Description

The **gpfs_stat_inode64()** subroutine is used to seek the specified inode and to retrieve that inode and its extended attributes from the inode scan. This subroutine combines **gpfs_seek_inode64()** and **get_next_inode64()**, but will only return the specified inode.

The data returned by **gpfs_next_inode64()** is overwritten by subsequent calls to **gpfs_next_inode64()**, **gpfs_seek_inode64()**, or **gpfs_stat_inode64()**.

The **termIno** parameter provides a way to partition an inode scan so it can be run on more than one node. It is only used by this call to control prefetching.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to an inode scan descriptor.

ino

The inode number to be returned.

termIno

Prefetches inodes up to this inode. The caller might specify **maxIno** from **gpfs_open_inodescan()** or 0 to allow prefetching over the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

Exit status

If the **gpfs_stat_inode64()** subroutine is successful, it returns a value of 0 and the **iattr** parameter is set to point to **gpfs_iattr_t**.

If the **gpfs_stat_inode64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EPERM

The caller must have superuser privilege.

ENOSYS

The **gpfs_stat_inode()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

ENOMEM

The buffer is too small.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_stat_inode_with_xattrs()

gpfs_stat_inode_with_xattrs() subroutine

Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_stat_inode_with_xattrs(gpfs_iscan_t *iscan,
                               gpfs_ino_t ino,
                               gpfs_ino_t termIno,
                               const gpfs_iattr_t **iattr,
                               const char **xattrBuf,
                               unsigned int *xattrBufLen);
```

Description

The **gpfs_stat_inode_with_xattrs()** subroutine is used to seek the specified inode and to retrieve that inode and its extended attributes from the inode scan. This subroutine combines **gpfs_seek_inode()** and **get_next_inode()**, but will only return the specified inode.

The data returned by **gpfs_next_inode()** is overwritten by subsequent calls to **gpfs_next_inode()**, **gpfs_seek_inode()**, or **gpfs_stat_inode_with_xattrs()**.

The **termIno** parameter provides a way to partition an inode scan such that it can be run on more than one node. It is only used by this call to control prefetching.

The returned values for **xattrBuf** and **xattrBufLen** must be provided to **gpfs_next_xattr()** to obtain the extended attribute names and values. The buffer used for the extended attributes is overwritten by subsequent calls to **gpfs_next_inode()**, **gpfs_seek_inode()**, or **gpfs_stat_inode_with_xattrs()**.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to an inode scan descriptor.

ino

The inode number to be returned.

termIno

Prefetches inodes up to this inode. The caller might specify **maxIno** from **gpfs_open_inodescan()** or 0 to allow prefetching over the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

xattrBuf

A pointer to the returned pointer to the **xattr** buffer.

xattrBufLen

The returned length of the **xattr** buffer.

Exit status

If the **gpfs_stat_inode_with_xattrs()** subroutine is successful, it returns a value of 0 and the **iattr** parameter is set to point to **gpfs_iattr_t**. If the **gpfs_stat_inode_with_xattrs()** subroutine is successful, but there are no more inodes before the **termIno** parameter, or if the requested inode does not exist, it returns a value of 0 and the **iattr** parameter is set to NULL.

If the **gpfs_stat_inode_with_xattrs()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EPERM

The caller must have superuser privilege.

ENOSYS

The **gpfs_stat_inode_with_xattrs()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

ENOMEM

The buffer is too small.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfs_stat_inode_with_xattrs64() subroutine

Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Synopsis

```
#include <gpfs.h>
int gpfs_stat_inode_with_xattrs64(gpfs_iscan_t *iscan,
                                gpfs_ino64_t ino,
                                gpfs_ino64_t termIno,
                                const gpfs_iattr64_t **iattr,
                                const char **xattrBuf,
                                unsigned int *xattrBufLen);
```

Description

The **gpfs_stat_inode_with_xattrs64()** subroutine is used to seek the specified inode and to retrieve that inode and its extended attributes from the inode scan. This subroutine combines **gpfs_seek_inode64()** and **get_next_inode64()**, but will only return the specified inode.

The data returned by **get_next_inode64()** is overwritten by subsequent calls to **gpfs_next_inode64()**, **gpfs_seek_inode64()**, or **gpfs_stat_inode_with_xattrs64()**.

The **termIno** parameter provides a way to partition an inode scan so it can be run on more than one node. It is only used by this call to control prefetching.

The returned values for **xattrBuf** and **xattrBufLen** must be provided to **gpfs_next_xattr()** to obtain the extended attribute names and values. The buffer used for the extended attributes is overwritten by subsequent calls to **gpfs_next_inode64()**, **gpfs_seek_inode64()**, or **gpfs_stat_inode_with_xattrs64()**.

Note: Compile any program that uses this subroutine with the **-lgpfs** flag from the following library:

- libgpfs.a for AIX
- libgpfs.so for Linux

Parameters

iscan

A pointer to an inode scan descriptor.

ino

The inode number to be returned.

termIno

Prefetches inodes up to this inode. The caller might specify **maxIno** from **gpfs_open_inodescan64()** or 0 to allow prefetching over the entire inode file.

iattr

A pointer to the returned pointer to the file's **iattr**.

xattrBuf

A pointer to the returned pointer to the **xattr** buffer.

xattrBufLen

The returned length of the **xattr** buffer.

Exit status

If the **gpfs_stat_inode_with_xattrs64()** subroutine is successful, it returns a value of 0 and the **iattr** parameter is set to point to **gpfs_iattr_t**. If the **gpfs_stat_inode_with_xattrs64()** subroutine is successful, but there are no more inodes before the **termIno** parameter, or if the requested inode does not exist, it returns a value of 0 and the **iattr** parameter is set to NULL.

If the **gpfs_stat_inode_with_xattrs64()** subroutine is unsuccessful, it returns a value of -1 and sets the global error variable **errno** to indicate the nature of the error.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

EPERM

The caller must have superuser privilege.

ENOSYS

The **gpfs_stat_inode_with_xattrs64()** subroutine is not supported under the current file system format.

ESTALE

The cached file system information was not valid.

ENOMEM

The buffer is too small.

GPFS_E_INVALID_ISCAN

Incorrect parameters.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsFcntlHeader_t

gpfsFcntlHeader_t structure

Contains declaration information for the `gpfs_fcntl()` subroutine.

Library

GPFS Library (`libgpfs.a` for AIX, `libgpfs.so` for Linux)

Structure

```
typedef struct
{
    int    totalLength;
    int    fcntlVersion;
    int    errorOffset;
    int    fcntlReserved;
} gpfsFcntlHeader_t;
```

Description

The `gpfsFcntlHeader_t` structure contains size, version, and error information for the `gpfs_fcntl()` subroutine.

Members

`totalLength`

This field must be set to the total length, in bytes, of the data structure being passed in this subroutine. This includes the length of the header and all hints and directives that follow the header.

The total size of the data structure *cannot* exceed the value of `GPFS_MAX_FCNTL_LENGTH`, as defined in the header file `gpfs_fcntl.h`. The current value of `GPFS_MAX_FCNTL_LENGTH` is 64 KB.

`fcntlVersion`

This field must be set to the current version number of the `gpfs_fcntl()` subroutine, as defined by `GPFS_FCNTL_CURRENT_VERSION` in the header file `gpfs_fcntl.h`. The current version number is one.

`errorOffset`

If an error occurs processing a system call, GPFS sets this field to the offset within the parameter area where the error was detected.

For example,

1. An incorrect version number in the header would cause `errorOffset` to be set to zero.
2. An error in the first hint following the header would set `errorOffset` to `sizeof(header)`.

If no errors are found, GPFS does not alter this field.

`fcntlReserved`

This field is currently unused.

For compatibility with future versions of GPFS, set this field to zero.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

gpfsGetFilesetName_t structure

Obtains the fileset name of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    char buffer[GPFS_FCNTL_MAX_NAME_BUFFER];
} gpfsGetFilesetName_t;
```

Description

The `gpfsGetFilesetName_t` structure is used to obtain a file's fileset name.

Members

structLen

Length of the `gpfsGetFilesetName_t` structure.

structType

Structure identifier `GPFS_FCNTL_GET_FILESETNAME`.

buffer

The size of the buffer may vary, but must be a multiple of eight. Upon successful completion of the call, the buffer contains a null-terminated character string for the name of the requested object.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsGetReplication_t structure

Obtains the replication factors of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int metadataReplicas;
    int maxMetadataReplicas;
    int dataReplicas;
    int maxDataReplicas;
    int status;
    int reserved;
} gpfsGetReplication_t;
```

Description

The **gpfsGetReplication_t** structure is used to obtain a file's replication factors.

Members

structLen

Length of the **gpfsGetReplication_t** structure.

structType

Structure identifier **GPFS_FCNTL_GET_REPLICATION**.

metadataReplicas

Returns the current number of copies of indirect blocks for the file.

maxMetadataReplicas

Returns the maximum number of copies of indirect blocks for a file.

dataReplicas

Returns the current number of copies of the data blocks for a file.

maxDataReplicas

Returns the maximum number of copies of data blocks for a file.

status

Returns the status of the file.

reserved

Unused, but should be set to 0.

Error status

These values are returned in the **status** field:

GPFS_FCNTL_STATUS_EXPOSED

This file may have some data where the only replicas are on suspended disks; implies some data may be lost if suspended disks are removed.

GPFS_FCNTL_STATUS_ILLREPLICATE

This file may not be properly replicated; that is, some data may have fewer or more than the desired number of replicas, or some replicas may be on suspended disks.

GPFS_FCNTL_STATUS_UNBALANCED

This file may not be properly balanced.

GPFS_FCNTL_STATUS_DATAUPDATEMISS

This file has stale data blocks on at least one of the disks that are marked as unavailable or recovering.

GPFS_FCNTL_STATUS_METAUPDATEMISS

This file has stale indirect blocks on at least one unavailable or recovering disk.

GPFS_FCNTL_STATUS_ILLPLACED

This file may not be properly placed; that is, some data may be stored in an incorrect storage pool.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsGetSetXAttr_t structure

Obtains or sets extended attribute values.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int nameLen;
    int bufferLen;
    unsigned int flags;
    int errReasonCode;
    char buffer[0];
} gpfsGetSetXAttr_t;
```

Description

The `gpfsGetSetXAttr_t` structure is used to obtain extended attributes.

Members

structLen

Length of the `gpfsGetSetXAttr_t` structure.

structType

Structure identifier `GPFS_FCNTL_GET_XATTR` or `GPFS_FCNTL_SET_XATTR`.

nameLen

Length of the attribute name. May include a trailing '\0' character.

bufferLen

For `GPFS_FCNTL_GET_XATTR`: Input, length of the buffer; output, length of the attribute value.

For `GPFS_FCNTL_SET_XATTR`: Input, length of the attribute value. Specify `-1` to delete an attribute.

errReasonCode

Reason code.

flags

The following flags are recognized:

- `GPFS_FCNTL_XATTRFLAG_NONE`
- `GPFS_FCNTL_XATTRFLAG_SYNC`
- `GPFS_FCNTL_XATTRFLAG_CREATE`
- `GPFS_FCNTL_XATTRFLAG_REPLACE`
- `GPFS_FCNTL_XATTRFLAG_DELETE`
- `GPFS_FCNTL_XATTRFLAG_NO_CTIME`
- `GPFS_FCNTL_XATTRFLAG_RESERVED`

buffer

Buffer for the attribute name and value.

For `GPFS_FCNTL_GET_XATTR`:

Input: The name begins at offset 0 and must be null terminated.

Output: The name is returned unchanged; the value begins at `nameLen` rounded up to a multiple of 8.

For **GPFS_FCNTL_SET_XATTR**:

Input: The name begins at offset 0 and must be null terminated. The value begins at **nameLen** rounded up to a multiple of 8.

The actual length of the buffer should be **nameLen** rounded up to a multiple of 8, plus the length of the attribute value rounded up to a multiple of 8.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsGetSnapshotName_t

gpfsGetSnapshotName_t structure

Obtains the snapshot name of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    char buffer[GPFS_FCNTL_MAX_NAME_BUFFER];
} gpfsGetSnapshotName_t;
```

Description

The `gpfsGetSnapshotName_t` structure is used to obtain a file's snapshot name. If the file is not part of a snapshot, a zero length snapshot name will be returned.

Members

structLen

Length of the `gpfsGetSnapshotName_t` structure.

structType

Structure identifier `GPFS_FCNTL_GET_SNAPSHOTNAME`.

buffer

The size of the buffer may vary, but must be a multiple of eight. Upon successful completion of the call, the buffer contains a null-terminated character string for the name of the requested object.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsGetStoragePool_t structure

Obtains the storage pool name of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    char buffer[GPFS_FCNTL_MAX_NAME_BUFFER];
} gpfsGetStoragePool_t;
```

Description

The `gpfsGetStoragePool_t` structure is used to obtain a file's storage pool name.

Members

structLen

Length of the `gpfsGetStoragePool_t` structure.

structType

Structure identifier `GPFS_FCNTL_GET_STORAGEPOOL`.

buffer

The size of the buffer may vary, but must be a multiple of eight. Upon successful completion of the call, the buffer contains a null-terminated character string for the name of the requested object.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsListXAttr_t

gpfsListXAttr_t structure

Lists extended attributes.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int bufferLen;
    int errReasonCode;
    char buffer[0];
} gpfsListXAttr_t;
```

Description

The `gpfsListXAttr_t` structure is used to list extended attributes.

Members

structLen

Length of the `gpfsListXAttr_t` structure.

structType

Structure identifier `GPFS_FCNTL_LIST_XATTR`.

bufferLen

Input: Length of the buffer. Output: Length of the returned list of names.

The actual length of the buffer required depends on the number of attributes set and the length of each attribute name. If the buffer provided is too small for all of the returned names, the **errReasonCode** will be set to `GPFS_FCNTL_ERR_BUFFER_TOO_SMALL`, and **bufferLen** will be set to the minimum size buffer required to list all attributes. An initial buffer length of 0 may be used to query the attributes and determine the correct buffer size for this file.

errReasonCode

Reason code.

buffer

Buffer for the returned list of names. Each attribute name is prefixed with a one-byte name length that includes the trailing null. The next attribute name follows immediately in the buffer (and is prefixed with its own length). Following the last name, a `'\0'` is appended to terminate the list. The returned **bufferLen** includes the final `'\0'`.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsRestripeData_t structure

Restripes the data blocks of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int options;
    int errReason;
    int errValue1;
    int errValue2;
    int reserved1;
    int reserved2;
} gpfsRestripeData_t;
```

Description

The **gpfsRestripeData_t** structure is used to restripe a file's data blocks to updates its replication and migrate its data. The data movement is always done immediately.

Members

structLen

Length of the **gpfsRestripeData_t** structure.

structType

Structure identifier **GPFS_FCNTL_RESTRIPE_DATA**.

options

Options for restripe command. See the **mmrestripefile** command for complete definitions.

GPFS_FCNTL_RESTRIPE_M

Migrate critical data off of suspended disks.

GPFS_FCNTL_RESTRIPE_R

Replicate data against subsequent failure.

GPFS_FCNTL_RESTRIPE_P

Place file data in assigned storage pool.

GPFS_FCNTL_RESTRIPE_B

Rebalance file data.

errReason

Reason code describing the failure. Possible codes are defined in "Error status" on page 872.

errValue1

Returned value depending upon **errReason**.

errValue2

Returned value depending upon **errReason**.

reserved1

Unused, but should be set to 0.

reserved2

Unused, but should be set to 0.

gpfsRestripeData_t

Error status

These values are returned in the **errReason** field:

GPFS_FCNTL_ERR_NO_REPLICA_GROUP

Not enough replicas could be created because the desired degree of replication is larger than the number of failure groups.

GPFS_FCNTL_ERR_NO_REPLICA_SPACE

Not enough replicas could be created because there was not enough space left in one of the failure groups.

GPFS_FCNTL_ERR_NO_BALANCE_SPACE

There was not enough space left on one of the disks to properly balance the file according to the current stripe method.

GPFS_FCNTL_ERR_NO_BALANCE_AVAILABLE

The file could not be properly balanced because one or more disks are unavailable.

GPFS_FCNTL_ERR_ADDR_BROKEN

All replicas were on disks that have since been deleted from the stripe group.

GPFS_FCNTL_ERR_NO_IMMUTABLE_DIR

No immutable attribute can be set on directories.

GPFS_FCNTL_ERR_NO_IMMUTABLE_SYSFILE

No immutable attribute can be set on system files.

GPFS_FCNTL_ERR_IMMUTABLE_FLAG

Immutable and indefinite retention flag is wrong.

GPFS_FCNTL_ERR_IMMUTABLE_PERM

Immutable and indefinite retention flag is wrong.

GPFS_FCNTL_ERR_APPENDONLY_CONFLICT

The **appendOnly** flag should be set separately.

GPFS_FCNTL_ERR_NOIMMUTABLE_ONSNAP

Cannot set immutable or **appendOnly** on snapshots.

GPFS_FCNTL_ERR_FILE_HAS_XATTRS

An attempt to change **maxDataReplicas** or **maxMetadataReplicas** was made on a file that has extended attributes.

GPFS_FCNTL_ERR_NOT_GPFS_FILE

This file is not part of a GPFS file system.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsSetReplication_t structure

Sets the replication factors of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int metadataReplicas;
    int maxMetadataReplicas;
    int dataReplicas;
    int maxDataReplicas;
    int errReason;
    int errValue1;
    int errValue2;
    int reserved;
} gpfsSetReplication_t;
```

Description

The **gpfsGetReplication_t** structure is used to set a file's replication factors. However, the directive does not cause the file to be restriped immediately. Instead, the caller must append a **gpfsRestripeData_t** directive or invoke an explicit restripe using the **mmrestripefs** or **mmrestripefile** command.

Members

structLen

Length of the **gpfsSetReplication_t** structure.

structType

Structure identifier **GPFS_FCNTL_SET_REPLICATION**.

metadataReplicas

Specifies how many copies of the file system's metadata to create. Enter a value of 1 or 2, but not greater than the value of the **maxMetadataReplicas** attribute of the file. A value of 0 indicates not to change the current value.

maxMetadataReplicas

The maximum number of copies of indirect blocks for a file. Space is reserved in the inode for all possible copies of pointers to indirect blocks. Valid values are 1 and 2, but cannot be less than **DefaultMetadataReplicas**. The default is 1. A value of 0 indicates not to change the current value.

dataReplicas

Specifies how many copies of the file data to create. Enter a value of 1 or 2, but not greater than the value of the **maxDataReplicas** attribute of the file. A value of 0 indicates not to change the current value.

maxDataReplicas

The maximum number of copies of data blocks for a file. Space is reserved in the inode and indirect blocks for all possible copies of pointers to data blocks. Valid values are 1 and 2, but cannot be less than **DefaultDataReplicas**. The default is 1. A value of 0 indicates not to change the current value.

errReason

Reason code describing the failure. Possible codes are defined in "Error status" on page 874.

errValue1

Returned value depending upon **errReason**.

gpfsSetReplication_t

errValue2

Returned value depending upon **errReason**.

reserved

Unused, but should be set to 0.

Error status

These values are returned in the **errReason** field:

GPFS_FCNTL_ERR_NONE

Command was successful or no reason information was returned.

GPFS_FCNTL_ERR_METADATA_REPLICAS_RANGE

Field **metadataReplicas** is out of range. Fields **errValue1** and **errValue2** contain the valid lower and upper range boundaries.

GPFS_FCNTL_ERR_MAXMETADATA_REPLICAS_RANGE

Field **maxMetadataReplicas** is out of range. Fields **errValue1** and **errValue2** contain the valid lower and upper range boundaries.

GPFS_FCNTL_ERR_DATA_REPLICAS_RANGE

Field **dataReplicas** is out of range. Fields **errValue1** and **errValue2** contain the valid lower and upper range boundaries.

GPFS_FCNTL_ERR_MAXDATA_REPLICAS_RANGE

Field **maxDataReplicas** is out of range. Fields **errValue1** and **errValue2** contain the valid lower and upper range boundaries.

GPFS_FCNTL_ERR_FILE_NOT_EMPTY

An attempt to change **maxMetadataReplicas** or **maxDataReplicas** or both was made on a file that is not empty.

GPFS_FCNTL_ERR_REPLICAS_EXCEED_FGMAX

Field **metadataReplicas**, or **dataReplicas**, or both exceed the number of failure groups. Field **errValue1** contains the maximum number of metadata failure groups. Field **errValue2** contains the maximum number of data failure groups.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

gpfsSetStoragePool_t structure

Sets the assigned storage pool of a file.

Library

GPFS Library (libgpfs.a for AIX, libgpfs.so for Linux)

Structure

```
typedef struct {
    int structLen;
    int structType;
    int errReason;
    int errValue1;
    int errValue2;
    int reserved;
    char buffer[GPFS_FCNTL_MAX_NAME_BUFFER];
} gpfsSetStoragePool_t;
```

Description

The **gpfsSetStoragePool_t** structure is used to set a file's assigned storage pool. However, the directive does not cause the file data to be migrated immediately. Instead, the caller must append a **gpfsRestripeData_t** directive or invoke an explicit restripe with the **mmrestripefs** or **mmrestripefile** command. The caller must have su or root privileges to change a storage pool assignment.

Members

structLen

Length of the **gpfsSetStoragePool_t** structure.

structType

Structure identifier **GPFS_FCNTL_SET_STORAGEPOOL**.

errReason

Reason code describing the failure. Possible codes are defined in "Error status."

errValue1

Returned value depending upon **errReason**.

errValue2

Returned value depending upon **errReason**.

reserved

Unused, but should be set to 0.

buffer

The name of the storage pool for the file's data. Only user files may be reassigned to different storage pool. System files, including all directories, must reside in the system pool and may not be moved. The size of the buffer may vary, but must be a multiple of eight.

Error status

These values are returned in the **errReason** field:

GPFS_FCNTL_ERR_NONE

Command was successful or no reason information was returned.

GPFS_FCNTL_ERR_NOPERM

User does not have permission to perform the requested operation.

gpfsSetStoragePool_t

GPFS_FCNTL_ERR_INVALID_STORAGE_POOL

Invalid storage pool name was given.

GPFS_FCNTL_ERR_INVALID_STORAGE_POOL_TYPE

Invalid storage pool. File cannot be assigned to given pool.

GPFS_FCNTL_ERR_INVALID_STORAGE_POOL_ISDIR

Invalid storage pool. Directories cannot be assigned to given pool.

GPFS_FCNTL_ERR_INVALID_STORAGE_POOL_ISLNK

Invalid storage pool. System files cannot be assigned to given pool.

GPFS_FCNTL_ERR_INVALID_STORAGE_POOL_ISSYS

Invalid storage pool. System files cannot be assigned to given pool.

GPFS_FCNTL_ERR_STORAGE_POOL_NOTENABLED

File system has not been upgraded to support storage pools.

Location

/usr/lpp/mmfs/lib/libgpfs.a for AIX

/usr/lpp/mmfs/lib/libgpfs.so for Linux

Chapter 13. GPFS user exits

Apart from the user exits define by using the **mmaddcallback** command, GPFS provides three more user exits: **mmsdrbackup**, **nsddevices**, and **syncfsconfig**.

Table 30 summarizes the GPFS-specific user exits.

Table 30. GPFS user exits

User exit	Purpose
"mmsdrbackup user exit" on page 878	Performs a backup of the GPFS configuration data.
"nsddevices user exit" on page 879	Identifies local physical devices that are used as GPFS Network Shared Disks (NSDs).
"syncfsconfig user exit" on page 880	Keeps file system configuration data in replicated clusters synchronized.

mmsdrbackup

mmsdrbackup user exit

Performs a backup of the GPFS configuration data.

Description

The `/var/mmfs/etc/mmsdrbackup` user exit, when properly installed on the primary GPFS configuration server, is called asynchronously every time there is a change to the GPFS master configuration file. You can use this user exit to create a backup of the GPFS configuration data.

Read the sample file `/usr/lpp/mmfs/samples/mmsdrbackup.sample` for a detailed description on how to code and install this user exit.

The type of backup that is created depends on the configuration of the cluster:

- If the Cluster Configuration Repository (CCR) is enabled, then a CCR backup is created. This type of backup applies to IBM Spectrum Scale V4.2.0 or later.
- Otherwise, a mmsdrfs backup is created.

For more information about the CCR, see the help topic on the `mmcrcluster` command in the *IBM Spectrum Scale: Administration and Programming Reference*.

Parameters

The generation number of the most recent version of the GPFS configuration data.

Exit status

The `mmsdrbackup` user exit returns a value of zero.

Location

`/var/mmfs/etc`

nsddevices user exit

Identifies local physical devices that are used as GPFS Network Shared Disks (NSDs).

Description

The `/var/mmfs/etc/nsddevices` user exit, when properly installed, is invoked synchronously by the GPFS daemon during its disk discovery processing. The purpose of this procedure is to discover and verify the physical devices on each node that correspond to the disks previously defined to GPFS with the `mmcrnsd` command. The `nsddevices` user exit can be used to either replace or to supplement the disk discovery procedure of the GPFS daemon.

Read the sample file `/usr/lpp/mmfs/samples/nsddevices.sample` for a detailed description on how to code and install this user exit.

Parameters

None.

Exit status

The `nsddevices` user exit should return either zero or one.

When the `nsddevices` user exit returns a value of zero, the GPFS disk discovery procedure is bypassed.

When the `nsddevices` user exit returns a value of one, the GPFS disk discovery procedure is performed and the results are concatenated with the results from the `nsddevices` user exit.

Location

`/var/mmfs/etc`

syncfsconfig

syncfsconfig user exit

Keeps file system configuration data in replicated clusters synchronized.

Description

The `/var/mmfs/etc/syncfsconfig` user exit, when properly installed, will be synchronously invoked after each command that may change the configuration of a file system. Examples of such commands are: `mmadddisk`, `mmdeldisk`, `mmchfs`, and so forth. The `syncfsconfig` user exit can be used to keep the file system configuration data in replicated GPFS clusters automatically synchronized.

Read the sample file `/usr/lpp/mmfs/samples/syncfsconfig.sample` for a detailed description on how to code and install this user exit.

Parameters

None.

Exit status

The `syncfsconfig` user exit should always return a value of zero.

Location

`/var/mmfs/etc`

Chapter 14. Considerations for GPFS applications

Application design must consider the exceptions to the Open Group technical standards for the **stat()** system call and NFS V4 ACLs. Also, a technique to check if a file system is controlled by GPFS has been provided.

Consider the following:

- “Exceptions to Open Group technical standards”
- “Determining if a file system is controlled by GPFS”
- “GPFS exceptions and limitations to NFS V4 ACLs” on page 882

Exceptions to Open Group technical standards

GPFS is designed in a way that most applications written to the Open Group technical standard for file system calls can access the GPFS data without any modification. However, there are some exceptions.

Applications that depend on exact reporting of changes to the following fields returned by the **stat()** call may not work as expected:

1. **exact mtime**
2. **mtime**
3. **ctime**
4. **atime**

Providing exact support for these fields would require significant performance degradation to all applications executing on the system. These fields are guaranteed accurate when the file is closed.

These values will be accurate on a node right after it accesses or modifies a file, but may not be accurate for a short while when a file is accessed or modified on some other node.

If 'exact mtime' is specified for a file system (using the **mmcrfs** or **mmchfs** commands with the **-E yes** flag), the **mtime** and **ctime** values are always correct by the time the **stat()** call gives its answer. If 'exact mtime' is not specified, these values will be accurate after a couple of minutes, to allow the synchronization daemons to propagate the values to all nodes. Regardless of whether 'exact mtime' is specified, the **atime** value will be accurate after a couple of minutes, to allow for all the synchronization daemons to propagate changes.

Alternatively, you may use the GPFS calls, **gpfs_stat()** and **gpfs_fstat()** to return exact **mtime** and **atime** values.

The delayed update of the information returned by the **stat()** call also impacts system commands which display disk usage, such as **du** or **df**. The data reported by such commands may not reflect changes that have occurred since the last sync of the file system. For a parallel file system, a sync does not occur until all nodes have individually synchronized their data. On a system with no activity, the correct values will be displayed after the sync daemon has run on all nodes.

Determining if a file system is controlled by GPFS

A file system is controlled by GPFS if the **f_type** field in the **statfs** structure returned from a **statfs()** or **fstatfs()** call has the value 0x47504653, which is GPFS in ASCII.

This constant is in the `gpfs.h` file, with the name `GPFS_SUPER_MAGIC`. If an application includes `gpfs.h`, it can compare `f_type` to `GPFS_SUPER_MAGIC` to determine if the file system is controlled by GPFS.

GPFS exceptions and limitations to NFS V4 ACLs

GPFS has some exceptions and limitations for the NFS V4 ACLs.

Those exceptions and limitations include:

1. Alarm type ACL entries are not supported.
2. Audit type ACL entries are not supported.
3. Some types of access for which NFS V4 defines controls do not currently exist in GPFS. For these, ACL entries will be accepted and saved, but since there is no corresponding operation they will have no effect. These include `READ_NAMED`, `WRITE_NAMED`, and `SYNCHRONIZE`.

Note: Even if GPFS ignores these bits, the SMB service will enforce them on the protocol level.

4. AIX requires that `READ_ACL` and `WRITE_ACL` always be granted to the object owner. Although this contradicts *NFS Version 4 Protocol*, it is viewed that this is an area where users would otherwise erroneously leave an ACL that only privileged users could change. Since ACLs are themselves file attributes, `READ_ATTR` and `WRITE_ATTR` are similarly granted to the owner. Since it would not make sense to then prevent the owner from accessing the ACL from a non-AIX node, GPFS has implemented this exception everywhere.
5. AIX does not support the use of special name values other than `owner@`, `group@`, and `everyone@`. Therefore, these are the only valid special name values for use in GPFS NFS V4 ACLs as well.
6. NFS V4 allows ACL entries that grant users (or groups) permission to change the owner or owning group of the file (for example, with the `chown` command). For security reasons, GPFS now restricts this so that non-privileged users may only `chown` such a file to themselves (becoming the owner) or to a group that they are a member of.
7. Windows does not support NFS V4 ACLs.
8. If a file system is to be exported over NFS V4/Linux, then it *must* be configured to support POSIX ACLs (with the `-k all` or `-k posix` option of the `mmcrfs` command). This is because NFS V4 Linux servers only handle ACLs properly if they are stored in GPFS as POSIX ACLs. For more information, see “Linux ACLs and extended attributes.”
9. Concurrent Samba, AIX NFS servers, and GPFS Windows nodes in the cluster are allowed. NFS V4 ACLs can be stored in GPFS file systems using Samba exports, NFS V4 AIX servers, GPFS Windows nodes, `aclput`, and `mmputacl`. However, clients of Linux V4 servers will not be able to see these ACLs, just the permissions from the mode.

For more information about GPFS ACLs and NFS export, see Chapter 10, “Managing GPFS access control lists,” on page 187.

Linux ACLs and extended attributes

NFS V4 uses the existing POSIX ACLs and the extended attribute support in Linux that is supported by GPFS.

Although the NFS V4 protocol defines a richer ACL model similar to Windows ACLs, the Linux implementation maps those ACLs to POSIX ACLs before passing them to the underlying file system. This mapping is done in `nfsd`. This means that if you set an ACL from a client and then fetch it back, you will see that what the server returns is not exactly what you set. This is because the ACL you set was converted to a POSIX ACL and then back again.

NFS V4 ACLs are more fine-grained than POSIX ACLs, so the POSIX-to-NFS V4 translation is close to perfect, but the NFS V4-to-POSIX translation is not. The NFS V4 server attempts to err on the side of

mapping to a stricter ACL. There is a very small set of NFS V4 ACLs that the server rejects completely (for example, any ACL that attempts to explicitly DENY rights to read attributes), but otherwise, the server tries very hard to accept ACLs and map them as best it can.

ACLs that are set through AIX/NFS V4 and Windows nodes are written as NFS V4 ACLs to GPFS, whereas ACLs that are set through Linux/NFS V4 are written as POSIX ACLs to GPFS. Currently, GPFS does not provide an interface to convert on-disk NFS V4 ACLs to POSIX ACLs. This means that if ACLs are written through either AIX/NFS V4 or Windows, they cannot be read by Linux/NFS V4. In this case, a Linux NFS V4 server constructs an ACL from the permission mode bits only and ignores the ACL on the file.

General NFS V4 Linux exceptions and limitations

GPFS has some NFS V4 Linux exceptions and limitations.

Specifically, Windows-based NFS V4 clients are not supported with Linux NFS V4 servers because of their use of share modes. In particular, GPFS does not support:

- A Windows NFS V4 client talking to multinode GPFS, where one of the nodes runs Linux. Since Windows `open()` allows share modes to be specified, these can be mapped directly to the NFS V4 share modes and passed to the server.
- A multinode GPFS cluster exported over NFS V4 and Samba, where one of the nodes runs Linux. Since Samba can enforce share modes throughout the cluster, interoperability will be broken with NFS V4, which cannot.
- An application that can be written to directly invoke the NFS V4 protocol. Since POSIX does not allow share modes to be specified, this is the only way a UNIX (AIX or Linux) application can indicate share modes.
- Applications running over NFS V4 that map certain operations to share reservations. An AIX NFS V4 client maps the `open(F_EXCL)` call to an `open(SHARE_DENY_READ)` over NFS V4.

Considerations for the use of direct I/O (`O_DIRECT`)

A file opened in the `O_DIRECT` mode (“direct I/O”), GPFS transfers data directly between the user buffer and the file on the disk.

Using direct I/O may provide some performance benefits in the following cases:

- The file is accessed at random locations.
- There is no access locality.

Direct transfer between the user buffer and the disk can only happen if all of the following conditions are true:

- The number of bytes transferred is a multiple of 512 bytes.
- The file offset is a multiple of 512 bytes.
- The user memory buffer address is aligned on a 512-byte boundary.

When these conditions are *not* all true, the operation will still proceed but will be treated more like other normal file I/O, with the `O_SYNC` flag that flushes the dirty buffer to disk.

When these conditions *are* all true, the GPFS pagepool is not used because the data is transferred directly; therefore, an environment in which most of the I/O volume is due to direct I/O (such as in databases) will not benefit from a large pagepool. Note, however, that the pagepool still needs to be configured with an adequate size, or left at its default value, because the pagepool is also used to store file metadata (especially for the indirect blocks required for large files).

With direct I/O, the application is responsible for coordinating access to the file, and neither the overhead nor the protection provided by GPFS locking mechanisms plays a role. In particular, if two threads or nodes perform direct I/O concurrently on overlapping portions of the file, the outcome is undefined. For example, when multiple writes are made to the same file offsets, it is undetermined which of the writes will be on the file when all I/O is completed. In addition, if the file has data replication, it is not guaranteed that all the data replicas will contain the data from the same writer. That is, the contents of each of the replicas could diverge.

Even when the I/O requests are aligned as previously listed, in the following cases GPFS will not transfer the data directly and will revert to the slower buffered behavior:

- The write causes the file to increase in size.
- The write is in a region of the file that has been preallocated (via `gpfs_prealloc()`) but has not yet been written.
- The write is in a region of the file where a “hole” is present; that is, the file is sparse and has some unallocated regions.

When direct I/O requests are aligned but none of the previously listed conditions (that would cause the buffered I/O path to be taken) are present, handling is optimized this way: the request is completely handled in kernel mode by the GPFS kernel module, without the GPFS daemon getting involved. Any of the following conditions, however, will still result in the request going through the daemon:

- The I/O operation needs to be served by an NSD server.
- The file system has data replication. In the case of a write operation, the GPFS daemon is involved to produce the log records that ensure that the replica contents are identical (in case of a failure while writing the replicas to disk).
- The operation is performed on the Windows operating system.

Note that setting the `O_DIRECT` flag on an open file with `fcntl (fd, F_SETFL, [..])`, which may be allowed on Linux, is ignored in a GPFS file system.

Chapter 15. File system format changes between versions of GPFS

Every GPFS file system has a format version number associated with it. This version number corresponds to the on-disk data structures of the file system and is an indicator of the supported file system functionality.

The file system version number is assigned when the file system is first created, and is updated to the latest supported level after the file system is migrated using the **mmchfs -V** command.

The format version number for a file system can be displayed with the **mmlsfs -V** command. If a file system was created with an older GPFS release, new functionality that requires different on-disk data structures will not be enabled until you run the **mmchfs -V** command. In addition to **mmchfs -V**, certain new features may require you to additionally run the **mmmigratefs** command.

Note: The **-V** option cannot be used to make file systems created prior to GPFS 3.2.1.5 available to Windows nodes. Windows nodes can mount only file systems that are created with GPFS 3.2.1.5 or later.

The **mmchfs -V** option requires the specification of one of two values - **full** or **compat**:

- Specifying **mmchfs -V full** enables all of the new functionality that requires different on-disk data structures. After this command, nodes in remote clusters running an older GPFS version will no longer be able to mount the file system.

Running the **mmchfs -V full** gives you a warning similar to the following:

Note:

```
# mmchfs n03NsdOnFile36 -V full
You have requested that the file system be upgraded to
version 15.01 (4.2.0.0). This will enable new functionality but will
prevent you from using the file system with earlier releases of GPFS.
Do you want to continue?
```

- Specifying **mmchfs -V compat** enables only features that are backward compatible with nodes running GPFS 3.2. After this command, nodes in remote clusters running GPFS 3.2 or later will still be able to mount the file system, but nodes running GPFS versions 3.1 or older will not be able to mount the file system.

The current highest file system format version is 15.01. This is the version that is assigned to file systems created with IBM Spectrum Scale 4.2.0.0. The same version number will be assigned to older file systems after you run the **mmchfs -V full** command.

- After running **mmchfs -V full**, the file system will be able to support the following:
 - Quality of Service (QoS) function is enabled
 - Compression

If your current file system is at format level 14.20 (IBM Spectrum Scale 4.1.1), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
 - Enabling and disabling of quota management without unmounting the file system.
 - The use of fileset-level integrated archive manager (IAM) modes.
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 14.04 (GPFS 4.1.0.0), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support different block allocation map types on an individual storage-pool basis.
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 13.23 (GPFS 3.5.0.7), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
 - directory block sizes up to 256 KB (previous maximum was 32 KB)
 - directories will be able to reduce their size when files are removed
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 13.01 (GPFS 3.5.0.1), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
 - extended storage pool properties
 - File Placement Optimizer (FPO)
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 12.03 (GPFS 3.4), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
 - independent filesets and snapshots of individual independent filesets
 - active file management (AFM)
 - storing the data for small files in the inode
 - file clones (writable snapshots of a file)
 - policy language support for new attributes, variable names, and functions: OPTS clause for the SET POOL and RESTORE rules, encoding of path names via an ESCAPE clause for the EXTERNAL LIST and EXTERNAL POOL rules, GetEnv(), GetMMconfig(), SetXattr(), REGEX().
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 11.03 (GPFS 3.3), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
 - more than 2,147,483,648 files
 - fast extended attributes (which requires **mmmigratefs** to be run also)
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 10.00 (GPFS 3.2.0.0) or 10.01 (GPFS 3.2.1.5), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus the following:

- new maximum number of filesets in a file system (10000)
- new maximum for the number of hard links per object (2**32)
- improved quota performance for systems with large number of users
- policy language support for new attributes, variable names, and functions: MODE, INODE, NLINK, RDEVICE_ID, DEVICE_ID, BLOCKSIZE, GENERATION, XATTR(), ATTR_INTEGER(), and XATTR_FLOAT()

If your current file system is at format level 9.03 (GPFS 3.1), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- fine grain directory locking
- LIMIT clause on placement policies

If your current file system is at format level 8.00 (GPFS 2.3), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- storage pools
- filesets
- fileset quotas

If your current file system is at format level 7.00 (GPFS 2.2), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- NFS V4 access control lists
- new format for the internal allocation summary files

If your current file system is at format level 6.00 (GPFS 2.1), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus extended access control list entries (**-rwx** access mode bits).

The functionality described in this topic is only a subset of the functional changes introduced with the different GPFS releases. Functional changes that do not require changing the on-disk data structures are not listed here. Such changes are either immediately available when the new level of code is installed, or require running the **mmchconfig release=LATEST** command. For a complete list, see the “Summary of changes” on page xv.

Accessibility features for IBM Spectrum Scale

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in IBM Spectrum Scale:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are discernible by touch but do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

IBM Knowledge Center, and its related publications, are accessibility-enabled. The accessibility features are described in IBM Knowledge Center (www.ibm.com/support/knowledgecenter).

Keyboard navigation

This product uses standard Microsoft Windows navigation keys.

IBM and accessibility

See the IBM Human Ability and Accessibility Center (www.ibm.com/able) for more information about the commitment that IBM has to accessibility.

Notices

This information was developed for products and services that are offered in the USA.

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
United States of America*

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan*

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: 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 states 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 supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM Corporation
Dept. H6MA/Building 707
Mail Station P300
2455 South Road
Poughkeepsie, NY 12601-5400
USA*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

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 the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Intel is a trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of the Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM's Privacy Policy at <http://www.ibm.com/privacy> and IBM's Online Privacy Statement at <http://www.ibm.com/privacy/details> the section entitled "Cookies, Web Beacons and Other Technologies" and the "IBM Software Products and Software-as-a-Service Privacy Statement" at <http://www.ibm.com/software/info/product-privacy>.

Glossary

This glossary provides terms and definitions for IBM Spectrum Scale.

The following cross-references are used in this glossary:

- *See* refers you from a nonpreferred term to the preferred term or from an abbreviation to the spelled-out form.
- *See also* refers you to a related or contrasting term.

For other terms and definitions, see the IBM Terminology website (www.ibm.com/software/globalization/terminology) (opens in new window).

B

block utilization

The measurement of the percentage of used subblocks per allocated blocks.

C

cluster

A loosely-coupled collection of independent systems (nodes) organized into a network for the purpose of sharing resources and communicating with each other. See also *GPFS cluster*.

cluster configuration data

The configuration data that is stored on the cluster configuration servers.

cluster manager

The node that monitors node status using disk leases, detects failures, drives recovery, and selects file system managers. The cluster manager must be a quorum node. The selection of the cluster manager node favors the quorum-manager node with the lowest node number among the nodes that are operating at that particular time.

Note: The cluster manager role is not moved to another node when a node with a lower node number becomes active.

control data structures

Data structures needed to manage file data and metadata cached in memory.

Control data structures include hash tables and link pointers for finding cached data; lock states and tokens to implement distributed locking; and various flags and sequence numbers to keep track of updates to the cached data.

D

Data Management Application Program Interface (DMAPI)

The interface defined by the Open Group's XDSM standard as described in the publication *System Management: Data Storage Management (XDSM) API Common Application Environment (CAE) Specification C429*, The Open Group ISBN 1-85912-190-X.

deadman switch timer

A kernel timer that works on a node that has lost its disk lease and has outstanding I/O requests. This timer ensures that the node cannot complete the outstanding I/O requests (which would risk causing file system corruption), by causing a panic in the kernel.

dependent fileset

A fileset that shares the inode space of an existing independent fileset.

disk descriptor

A definition of the type of data that the disk contains and the failure group to which this disk belongs. See also *failure group*.

disk leasing

A method for controlling access to storage devices from multiple host systems. Any host that wants to access a storage device configured to use disk leasing registers for a lease; in the event of a perceived failure, a host system can deny access, preventing I/O operations with the storage device until the preempted system has reregistered.

disposition

The session to which a data management event is delivered. An individual disposition is set for each type of event from each file system.

domain

A logical grouping of resources in a network for the purpose of common management and administration.

E**ECKD™**

See *extended count key data (ECKD)*.

ECKD device

See *extended count key data device (ECKD device)*.

encryption key

A mathematical value that allows components to verify that they are in communication with the expected server. Encryption keys are based on a public or private key pair that is created during the installation process. See also *file encryption key*, *master encryption key*.

extended count key data (ECKD)

An extension of the count-key-data (CKD) architecture. It includes additional commands that can be used to improve performance.

extended count key data device (ECKD device)

A disk storage device that has a data transfer rate faster than some processors can utilize and that is connected to the processor through use of a speed matching buffer. A specialized channel program is needed to communicate with such a device. See also *fixed-block architecture disk device*.

F**failback**

Cluster recovery from failover following repair. See also *failover*.

failover

(1) The assumption of file system duties by another node when a node fails. (2) The process of transferring all control of the ESS to a single cluster in the ESS when the other clusters in the ESS fails. See also *cluster*. (3) The routing of all transactions to a second controller when the first controller fails. See also *cluster*.

failure group

A collection of disks that share common access paths or adapter connection, and could all become unavailable through a single hardware failure.

FEK See *file encryption key*.

fileset A hierarchical grouping of files managed as a unit for balancing workload across a cluster. See also *dependent fileset*, *independent fileset*.

fileset snapshot

A snapshot of an independent fileset plus all dependent filesets.

file clone

A writable snapshot of an individual file.

file encryption key (FEK)

A key used to encrypt sectors of an individual file. See also *encryption key*.

file-management policy

A set of rules defined in a policy file that GPFS uses to manage file migration and file deletion. See also *policy*.

file-placement policy

A set of rules defined in a policy file that GPFS uses to manage the initial placement of a newly created file. See also *policy*.

file system descriptor

A data structure containing key information about a file system. This information includes the disks assigned to the file system (*stripe group*), the current state of the file system, and pointers to key files such as quota files and log files.

file system descriptor quorum

The number of disks needed in order to write the file system descriptor correctly.

file system manager

The provider of services for all the nodes using a single file system. A file system manager processes changes to the state or description of the file system, controls the regions of disks that are allocated to each node, and controls token management and quota management.

fixed-block architecture disk device (FBA disk device)

A disk device that stores data in blocks of fixed size. These blocks are addressed by block number relative to the beginning of the file. See also *extended count key data device*.

fragment

The space allocated for an amount of data

too small to require a full block. A fragment consists of one or more subblocks.

G

global snapshot

A snapshot of an entire GPFS file system.

GPFS cluster

A cluster of nodes defined as being available for use by GPFS file systems.

GPFS portability layer

The interface module that each installation must build for its specific hardware platform and Linux distribution.

GPFS recovery log

A file that contains a record of metadata activity, and exists for each node of a cluster. In the event of a node failure, the recovery log for the failed node is replayed, restoring the file system to a consistent state and allowing other nodes to continue working.

I

ill-placed file

A file assigned to one storage pool, but having some or all of its data in a different storage pool.

ill-replicated file

A file with contents that are not correctly replicated according to the desired setting for that file. This situation occurs in the interval between a change in the file's replication settings or suspending one of its disks, and the restripe of the file.

independent fileset

A fileset that has its own inode space.

indirect block

A block containing pointers to other blocks.

inode The internal structure that describes the individual files in the file system. There is one inode for each file.

inode space

A collection of inode number ranges reserved for an independent fileset, which enables more efficient per-fileset functions.

ISKLM

IBM Security Key Lifecycle Manager. For GPFS encryption, the ISKLM is used as an RKM server to store MEKs.

J

journaled file system (JFS)

A technology designed for high-throughput server environments, which are important for running intranet and other high-performance e-business file servers.

junction

A special directory entry that connects a name in a directory of one fileset to the root directory of another fileset.

K

kernel The part of an operating system that contains programs for such tasks as input/output, management and control of hardware, and the scheduling of user tasks.

M

master encryption key (MEK)

A key used to encrypt other keys. See also *encryption key*.

MEK See *master encryption key*.

metadata

Data structures that contain information that is needed to access file data. Metadata includes inodes, indirect blocks, and directories. Metadata is not accessible to user applications.

metanode

The one node per open file that is responsible for maintaining file metadata integrity. In most cases, the node that has had the file open for the longest period of continuous time is the metanode.

mirroring

The process of writing the same data to multiple disks at the same time. The mirroring of data protects it against data loss within the database or within the recovery log.

multi-tailed

A disk connected to multiple nodes.

N

namespace

Space reserved by a file system to contain the names of its objects.

Network File System (NFS)

A protocol, developed by Sun Microsystems, Incorporated, that allows any host in a network to gain access to another host or netgroup and their file directories.

Network Shared Disk (NSD)

A component for cluster-wide disk naming and access.

NSD volume ID

A unique 16 digit hex number that is used to identify and access all NSDs.

node An individual operating-system image within a cluster. Depending on the way in which the computer system is partitioned, it may contain one or more nodes.

node descriptor

A definition that indicates how GPFS uses a node. Possible functions include: manager node, client node, quorum node, and nonquorum node.

node number

A number that is generated and maintained by GPFS as the cluster is created, and as nodes are added to or deleted from the cluster.

node quorum

The minimum number of nodes that must be running in order for the daemon to start.

node quorum with tiebreaker disks

A form of quorum that allows GPFS to run with as little as one quorum node available, as long as there is access to a majority of the quorum disks.

non-quorum node

A node in a cluster that is not counted for the purposes of quorum determination.

P

policy A list of file-placement, service-class, and encryption rules that define characteristics and placement of files. Several policies can be defined within the configuration, but only one policy set is active at one time.

policy rule

A programming statement within a policy that defines a specific action to be performed.

pool A group of resources with similar characteristics and attributes.

portability

The ability of a programming language to compile successfully on different operating systems without requiring changes to the source code.

primary GPFS cluster configuration server

In a GPFS cluster, the node chosen to maintain the GPFS cluster configuration data.

private IP address

A IP address used to communicate on a private network.

public IP address

A IP address used to communicate on a public network.

Q

quorum node

A node in the cluster that is counted to determine whether a quorum exists.

quota The amount of disk space and number of inodes assigned as upper limits for a specified user, group of users, or fileset.

quota management

The allocation of disk blocks to the other nodes writing to the file system, and comparison of the allocated space to quota limits at regular intervals.

R

Redundant Array of Independent Disks (RAID)

A collection of two or more disk physical drives that present to the host an image of one or more logical disk drives. In the event of a single physical device failure, the data can be read or regenerated from the other disk drives in the array due to data redundancy.

recovery

The process of restoring access to file system data when a failure has occurred. Recovery can involve reconstructing data or providing alternative routing through a different server.

remote key management server (RKM server)

A server that is used to store master encryption keys.

replication

The process of maintaining a defined set of data in more than one location. Replication involves copying designated changes for one location (a source) to another (a target), and synchronizing the data in both locations.

RKM server

See *remote key management server*.

rule A list of conditions and actions that are triggered when certain conditions are met. Conditions include attributes about an object (file name, type or extension, dates, owner, and groups), the requesting client, and the container name associated with the object.

S**SAN-attached**

Disks that are physically attached to all nodes in the cluster using Serial Storage Architecture (SSA) connections or using Fibre Channel switches.

Scale Out Backup and Restore (SOBAR)

A specialized mechanism for data protection against disaster only for GPFS file systems that are managed by Tivoli Storage Manager (TSM) Hierarchical Storage Management (HSM).

secondary GPFS cluster configuration server

In a GPFS cluster, the node chosen to maintain the GPFS cluster configuration data in the event that the primary GPFS cluster configuration server fails or becomes unavailable.

Secure Hash Algorithm digest (SHA digest)

A character string used to identify a GPFS security key.

session failure

The loss of all resources of a data management session due to the failure of the daemon on the session node.

session node

The node on which a data management session was created.

Small Computer System Interface (SCSI)

An ANSI-standard electronic interface that allows personal computers to

communicate with peripheral hardware, such as disk drives, tape drives, CD-ROM drives, printers, and scanners faster and more flexibly than previous interfaces.

snapshot

An exact copy of changed data in the active files and directories of a file system or fileset at a single point in time. See also *fileset snapshot*, *global snapshot*.

source node

The node on which a data management event is generated.

stand-alone client

The node in a one-node cluster.

storage area network (SAN)

A dedicated storage network tailored to a specific environment, combining servers, storage products, networking products, software, and services.

storage pool

A grouping of storage space consisting of volumes, logical unit numbers (LUNs), or addresses that share a common set of administrative characteristics.

stripe group

The set of disks comprising the storage assigned to a file system.

striping

A storage process in which information is split into blocks (a fixed amount of data) and the blocks are written to (or read from) a series of disks in parallel.

subblock

The smallest unit of data accessible in an I/O operation, equal to one thirty-second of a data block.

system storage pool

A storage pool containing file system control structures, reserved files, directories, symbolic links, special devices, as well as the metadata associated with regular files, including indirect blocks and extended attributes. The **system storage pool** can also contain user data.

T**token management**

A system for controlling file access in which each application performing a read or write operation is granted some form of access to a specific block of file data.

Token management provides data consistency and controls conflicts. Token management has two components: the token management server, and the token management function.

token management function

A component of token management that requests tokens from the token management server. The token management function is located on each cluster node.

token management server

A component of token management that controls tokens relating to the operation of the file system. The token management server is located at the file system manager node.

twin-tailed

A disk connected to two nodes.

U

user storage pool

A storage pool containing the blocks of data that make up user files.

V

VFS See *virtual file system*.

virtual file system (VFS)

A remote file system that has been mounted so that it is accessible to the local user.

virtual node (vnode)

The structure that contains information about a file system object in a virtual file system (VFS).

Index

Special characters

- aix-trace-buffer-size
 - changing 681
- trace-dispatch
 - changing 681
- tracedev-buffer-size
 - changing 681
- tracedev-compression-level
 - changing 681
- tracedev-overwrite-buffer-size
 - changing 682
- tracedev-write-mode
 - changing 682
- traceFileSize
 - changing 681

A

- access ACL 188
- access control information 773
 - restoring 841
- access control lists
 - administering 191
 - allow type 191
 - applying 194
 - authorize file protocol users 200
 - authorizing object users 209, 213
 - authorizing protocol users 200
 - limitations 214
 - best practices 202
 - change NFS V4 ACL 194
 - changing 190
 - creating 614
 - DELETE 193
 - DELETE_CHILD 193
 - deleting 190, 195, 446
 - deny type 191
 - display NFS V4 ACL 194
 - displaying 189, 500
 - editing 478
 - exceptions 195
 - export-level ACLs 201
 - getting 503
 - inheritance 191
 - DirInherit 192
 - FileInherit 192
 - Inherited 192
 - InheritOnly 192
 - inheritance flags 201
 - limitations 195
 - Linux 882
 - managing 187
 - NFS V4 187, 191
 - NFS V4 syntax 191
 - object ACLs 209
 - creating write ACLs 213
 - required permissions 203
 - setting 188, 189, 194
 - special names 192
 - traditional 187

- access control lists (*continued*)
 - translation 193
 - work with ACLs 207
- accessibility features for IBM Spectrum Scale 889
- ACL information 726, 826
 - restoring 841
 - retrieving 773
- activating quota limit checking 178
- active commands
 - listing 5
- Active Directory
 - authentication for file access 77
- AD for file
 - prerequisites 79
- AD for file authentication
 - AD with automatic ID mapping 80
 - AD with RFC2307 ID mapping 81
- AD-based authentication 83
- AD-based authentication for object access 95
 - AD with TLS 97
 - AD without TLS 96
- adding
 - disks 158, 239
- adding nodes to a GPFS cluster 8, 245
- additional calls, setup of interface for 814
- administering
 - GPFS file system 1, 2
- administration tasks 1, 2, 3, 4
- adminMode
 - requirements for administering GPFS 2
- adminMode attribute 333
- AFM 248, 251, 264, 611
- appendOnly file attribute 321, 519
- atime 373, 827, 830, 833, 836, 852, 854, 856, 881
- atimeDeferredSeconds attribute 336
- attributes
 - adminMode 2, 333
 - atimeDeferredSeconds 336
 - autoload 336
 - automountDir 336
 - cesSharedRoot 336
 - cipherList 336
 - cnfsGrace 337
 - cnfsMountdPort 337
 - cnfsNFSDprocs 337
 - cnfsReboot 337
 - cnfsSharedRoot 337
 - cnfsVersions 337
 - dataDiskWaitTimeForRecovery 337
 - deadlockBreakupDelay 338
 - deadlockDataCollectionDailyLimit 338
 - deadlockDataCollectionMinInterval 338
 - deadlockDetectionThreshold 338
 - deadlockDetectionThresholdForShortWaiters 338
 - deadlockDetectionThresholdIfOverloaded 338
 - deadlockOverloadThreshold 338
 - defaultHelperNodes 338
 - defaultMountDir 338
 - disableNodeUpdateOnFdatasync 339
 - dmapiDataEventRetry 339
 - dmapiEventTimeout 339

attributes (*continued*)

- dmapiMountEvent 339
- dmapiMountTimeout 340
- dmapiSessionFailureTimeout 340
- enableIPv6 340
- enforceFilesetQuotaOnRoot 340
- expelDataCollectionDailyLimit 341
- expelDataCollectionMinInterval 341
- failureDetectionTime 341
- fastestPolicyCmpThreshold 341
- fastestPolicyMaxValidPeriod 341
- fastestPolicyMinDiffPercent 341
- fastestPolicyNumReadSamples 341
- fileHeatLossPercent 341
- fileHeatPeriodMinutes 341
- FIPS1402mode 341
- forceLogWriteOnFdatasync 341
- lrocData 342
- lrocDataMaxFileSize 342
- lrocDataStubFileSize 342
- lrocDirectories 342
- lrocInodes 342
- maxblocksize 342
- maxDownDisksForRecovery 342
- maxFailedNodesForRecovery 343
- maxFcntlRangesPerFile 343
- maxFilesToCache 343
- maxMBpS 343
- maxStatCache 343
- metadataDiskWaitTimeForRecovery 343
- minDiskWaitTimeForRecovery 343
- mmapRangeLock 344
- nistCompliance 344
- noSpaceEventInterval 344
- nsdBufSpace 344
- nsdRAIDBufferPoolSizePct 344
- nsdRAIDTracks 344
- nsdServerWaitTimeForMount 345
- nsdServerWaitTimeWindowOnMount 345
- numaMemoryInterleave 345
- pagepool 345
- pagepoolMaxPhysMemPct 346
- prefetchThreads 346
- readReplicaPolicy 346
- release 347
- restripeOnDiskFailure 347
- rpcPerfNumberDayIntervals 347
- rpcPerfNumberHourIntervals 347
- rpcPerfNumberMinuteIntervals 347
- rpcPerfNumberSecondIntervals 348
- rpcPerfRawExecBufferSize 348
- rpcPerfRawStatBufferSize 348
- sidAutoMapRangeLength 348
- sidAutoMapRangeStart 348
- subnets 348
- systemLogLevel 349
- tiebreakerDisks 349
- uidDomain 349
- unmountOnDiskFail 349
- useNSDserver 166
- usePersistentReserve 350
- verbsPorts 350
- verbsRdma 351
- verbsRdmaCm 351
- verbsRdmaRoCEToS 351
- verbsRdmaSend 351
- verbsRdmPerConnection 351

attributes (*continued*)

- verbsRdmPerNode 351
- verbsSendBufferMemoryMB 351
- worker1Threads 352

authentication 77

- authentication for file access 69, 76
 - AD with automatic ID mapping 80
 - AD with RFC2307 ID mapping 81
 - LDAP-based authentication 84
 - NIS-based authentication 89
 - set up ID map range 78
- authentication for object access 69, 93
 - AD-based authentication 95
 - external Keystone server 100
 - LDAP-based authentication 98
 - local authentication 94
 - local authentication with SSL 95
- deleting 106
- limitations 109
- listing 107
- modifying 108
- protocol user authentication 69
 - set up authentication servers 70
- set up authentication servers
 - integrating with AD server 70
 - integrating with Keystone Identity Service 76
 - integrating with LDAP server 71
- User-defined method of authentication 89
- verifying 108

authentication considerations

- NFSv4 considerations 112

authentication limitations 109

authorizing protocol users 200

- authorize file protocol users 200
- authorizing object users 209, 213
- export-level ACLs 201
- limitations 214
- object ACLs
 - creating read ACLs 211
 - creating write ACLs 213
 - work with ACLs 207

autoload attribute 336

automated installation toolkit 711

automatic mount, indicating 418

automount 27

automountDir attribute 336

availability

- disk 162

B

backing up a file system 46, 49

- configuration information 290
- tuning with mmbackup 50
- using the GPFS policy engine 52
- using the mmbackup command 46

backing up a fileset 46

- using the mmbackup command 47

backing up a temporary snapshot to the TSM server 49

backing up file system configuration information

- using the mmbackupconfig command 52

backup applications

- writing 53

backup server 281

Base Tivoli Storage Manager 57

- best practices
 - configuring AD with RFC2307 as the authentication method 83
- BigInsights Hadoop distribution
 - mmhadoopctl 506
- bind user requirements 71
- block level incremental backups 810
- block size
 - choosing 418
 - effect on maximum mounted file system size 342, 418

C

- cache 198
- callbacks 226
 - daRebuildFailed 233
 - nsdChecksumMismatch 235
 - pdFailed 235
 - pdPathDown 235
 - pdRecovered 235
 - pdReplacePdisk 235
 - postRGRelinquish 236
 - postRGTakeover 236
 - preRGRelinquish 235
 - preRGTakeover 236
 - rgOpenFailed 236
 - rgPanic 237
- ces
 - config 663
 - mmsmb 663
- CES
 - configuration 63, 112, 304, 570
 - file systems 66
 - filesets 66
 - nodes 64
 - protocol service IP addresses 64
 - shared root file system 63
 - verification 66
 - mmces 304
 - mmcesdr 313
 - mmnfs 570
 - mmobj 581
 - mmprotocoltrace 607
 - mmuserauth 690
 - multiprotocol exports 121
 - NFS export configuration
 - changing 120
 - create 120
 - NFS exports
 - removal 121
 - Object protocol services
 - starting 68
 - protocol services
 - disabling 69
 - protocol tracing 607
 - SMB and NFS protocol services
 - starting 67
 - SMB configuration
 - export ACL 114
 - exports 113
 - SMB export configuration
 - changing 114
 - SMB exports
 - removal 114
 - SMB limitations
 - exports 119
 - topic 690

- CES packages
 - deploying 65
- cesSharedRoot attribute 336
- changing
 - an administration or daemon interface for a node 381
- attributes
 - adminMode 333
 - atimeDeferredSeconds 336
 - autoload 336
 - automountDir 336
 - cesSharedRoot 336
 - cipherList 336
 - cluster configuration 331
 - cnfsGrace 337
 - cnfsMountdPort 337
 - cnfsNFSDprocs 337
 - cnfsReboot 337
 - cnfsSharedRoot 337
 - cnfsVersions 337
 - dataDiskWaitTimeForRecovery 337
 - deadlockBreakupDelay 338
 - deadlockDataCollectionDailyLimit 338
 - deadlockDataCollectionMinInterval 338
 - deadlockDetectionThreshold 338
 - deadlockDetectionThresholdForShortWaiters 338
 - deadlockDetectionThresholdIfOverloaded 338
 - deadlockOverloadThreshold 338
 - defaultHelperNodes 338
 - defaultMountDir 338
 - disableNodeUpdateOnFdatasync 339
 - dmapiDataEventRetry 339
 - dmapiEventTimeout 339
 - dmapiMountEvent 339
 - dmapiMountTimeout 340
 - dmapiSessionFailureTimeout 340
 - enableIPv6 340
 - enforceFilesetQuotaOnRoot 340
 - expelDataCollectionDailyLimit 341
 - expelDataCollectionMinInterval 341
 - failureDetectionTime 341
 - fastestPolicyCmpThreshold 341
 - fastestPolicyMaxValidPeriod 341
 - fastestPolicyMinDiffPercent 341
 - fastestPolicyNumReadSamples 341
 - fileHeatLossPercent 341
 - fileHeatPeriodMinutes 341
 - FIPS1402mode 341
 - forceLogWriteOnFdatasync 341
 - IrocData 342
 - IrocDataMaxFileSize 342
 - IrocDataStubFileSize 342
 - IrocDirectories 342
 - IrocInodes 342
 - maxblocksize 342
 - maxDownDisksForRecovery 342
 - maxFailedNodesForRecovery 343
 - maxFcntlRangesPerFile 343
 - maxFilesToCache 343
 - maxMBpS 343
 - maxStatCache 343
 - metadataDiskWaitTimeForRecovery 343
 - minDiskWaitTimeForRecovery 343
 - mmapRangeLock 344
 - nistCompliance 344
 - noSpaceEventInterval 344
 - nsdBufSpace 344
 - nsdRAIDBufferPoolSizePct 344

- changing (*continued*)
 - attributes (*continued*)
 - nsdRAIDTracks 344
 - nsdServerWaitTimeForMount 345
 - nsdServerWaitTimeWindowOnMount 345
 - numaMemoryInterleave 345
 - pagepool 345
 - pagepoolMaxPhysMemPct 346
 - prefetchThreads 346
 - readReplicaPolicy 346
 - release 347
 - restripeOnDiskFailure 347
 - rpcPerfNumberDayIntervals 347
 - rpcPerfNumberHourIntervals 347
 - rpcPerfNumberMinuteIntervals 347
 - rpcPerfNumberSecondIntervals 348
 - rpcPerfRawExecBufferSize 348
 - rpcPerfRawStatBufferSize 348
 - sidAutoMapRangeLength 348
 - sidAutoMapRangeStart 348
 - subnets 348
 - systemLogLevel 349
 - tiebreakerDisks 349
 - uidDomain 349
 - unmountOnDiskFail 349
 - usePersistentReserve 350
 - verbsPorts 350
 - verbsRdma 351
 - verbsRdmaCm 351
 - verbsRdmaRoCEToS 351
 - verbsRdmaSend 351
 - verbsRdmPerConnection 351
 - verbsRdmPerNode 351
 - verbsSendBufferMemoryMB 351
 - worker1Threads 352
 - configuration attributes on the mmchconfig command 11
 - disk parameters 354
 - disk states 163, 354
 - fileset attributes 365
 - quotas 173
 - replication 34, 35
 - tracing attributes 680
 - user-defined node classes 385
- changing Quality of Service for I/O operations (QoS)
 - level 396
- changing quota limit checking 179
- changing storage pool properties 394
- checking
 - file systems 30
 - quotas 176
- chmod 188
- cipherList attribute 277, 336
- cleanup after GPFS interface calls 815
- Client license 377
- client node
 - refresh NSD server 579
- clone, file
 - copy 400, 727
 - decloning 738
 - redirect 400
 - show 400
 - snap 400, 729
 - split 400, 731
 - unsnap 733
- cluster
 - changing configuration attributes 11, 331
 - changing tracing attributes 680

- cluster (*continued*)
- configuration data 485
- cluster configuration attributes
 - changing 11
 - displaying 526
- cluster configuration data 497
- cluster configuration server 403, 496
- Cluster Export Services
 - config 663
 - configuration 304, 570
 - mmces 304
 - mmcesdr 313
 - mmnfs 570
 - mmobj 581
 - mmprotocoltrace 607
 - mmsmb 663
 - mmuserauth 690
 - protocol tracing 607
 - topic 690
- clustered NFS subsystem
 - using 199
- CNFS 199
- cnfsGrace attribute 337
- cnfsMountdPort attribute 337
- cnfsNFSDprocs attribute 337
- cnfsReboot attribute 337
- cnfsSharedRoot attribute 337
- cnfsVersions attribute 337
- commands 217
 - active 5
 - chmod 188
 - gpfs.snap 221
 - mmaddcallback 226
 - mmadddisk 158, 239, 880
 - mmaddnode 8, 245
 - mmafmconfig 248
 - mmafmctl 251
 - mmafmlocal 264
 - mmapplypolicy 46, 266
 - mmauth 20, 276
 - mmbackup 46, 47, 48, 49, 50, 51, 281
 - mmbackupconfig 46, 290
 - mmbuildgpl 292
 - mmcallhome 293
 - mmces 304
 - mmcesdr 313
 - mmchattr 34, 35, 42, 321
 - mmchcluster 10, 327
 - mmchconfig 11, 19, 20, 23, 197, 198, 331
 - mmchdisk 32, 42, 162, 163, 354
 - mmcheckquota 30, 169, 176, 179, 361
 - mmchfileset 365
 - mmchfs 34, 166, 169, 178, 179, 198, 371
 - mmchlicense 377
 - mmchmgr 379
 - mmchnode 381
 - mmchnodeclass 385
 - mmchnsd 165, 388
 - mmchpolicy 391
 - mmchpool 394
 - mmchqos 40, 396
 - mmclone 400
 - mmcrcluster 7, 403
 - mmcrfileset 408
 - mmcrfs 27, 169, 178, 191, 198, 414
 - mmcrnodeclass 424
 - mmcrnsd 157, 158, 239, 426, 648, 879, 880

commands (*continued*)

- mmdcrsnapshot 53, 431
- mmdefedquota 434
- mmdefquotaoff 437
- mmdefquotaon 440
- mmdefragfs 44, 45, 443
- mmdeacl 190, 191, 195, 446
- mmdelcallback 448
- mmdeldisk 159, 449, 880
- mmdelfileset 455
- mmdelfs 29, 458
- mmdelnode 9, 460
- mmdelnodeclass 463
- mmdelnsd 465
- mmdelsnapshot 467
- mmdf 43, 159, 470
- mmdiag 473
- mmeditacl 190, 191, 193, 194, 478
- mmedquota 169, 173, 481
- mmexportfs 485
- mmfsck 30, 32, 159, 487
- mmfsctl 496
- mmgetacl 188, 189, 193, 194, 195, 500
- mmgetstate 503
- mmhadoopctl 506
- mmimgbackup 508
- mmimgrestore 511
- mmimportfs 513
- mmlinkfileset 517
- mmlsattr 34, 519
- mmlscallback 522
- mmlscluster 7, 524
- mmlsconfig 526
- mmlsdisk 32, 162, 528
- mmlsfileset 532
- mmlsfs 33, 162, 178, 179, 198, 536
- mmlslicense 540
- mmlsmgr 24, 542
- mmlsmount 30, 544
- mmlsnodeclass 546
- mmlsnsd 157, 549
- mmlspolicy 552
- mmlspool 554
- mmlsqos 40, 556
- mmlsquota 177, 559
- mmlssnapshot 563, 777, 781
- mmmigratefs 566
 - fastea 566
- mmm mount 27, 28, 166, 568
- mmnfs 570
- mmnsdiscover 579
- mmobj 581
- mmperfmon query 592
- mmppmon 602
- mmprotocoltrace 607
- mmpsnap 611
- mmputacl 188, 189, 191, 194, 195, 614
- mmquotaoff 178, 179, 617
- mmquotaon 178, 619
- mmremotecluster 621
- mmremotefs 166, 624
- mmrepquota 179, 627
- mmrestoreconfig 631
- mmrestorefs 635
- mmrestripefile 639
- mmrestripefs 42, 43, 159, 162, 642
 - completion time 24

commands (*continued*)

- mmrpldisk 161, 648
- mmsdrrestore 655
- mmsetquota 169, 657
- mmshutdown 26, 661
- mmsmb 663
- mmsnapdir 674, 754
- mmstartup 25, 678
- mmtracectl 680
- mmumount 29, 684
- mmunlinkfileset 687
- mmuserauth 69, 76, 690
- mmwinservctl 709
- spectrumscale 711
- common GPFS command principles 3, 4
- configuration
 - ID mapping 106
- configuration attributes on the mmchconfig command
 - changing 11
- configuration tasks
 - CES 63, 112
 - CES nodes 64
 - CES protocol
 - service IP addresses 64
 - CES shared root file system 63
 - CES verification 66
 - changing NFS exports 120
 - changing SMB exports 114
 - disabling protocol services 69
 - file systems 66
 - filesets 66
 - NFS export removal 121
 - setting quotas 174
 - SMB and NFS protocols 121
 - SMB export ACL creation 114
 - SMB export creation 113
 - SMB export limitations 119
 - SMB export removal 114
- configured services 108
- Configuring
 - with LDAP ID mapping 83
- configuring AD with RFC2307 83
- configuring AD with RFC2307 as the authentication method
 - best practices 83
- connector for Hadoop distributions, GPFS
 - mmhadoopctl 506
- considerations for changing
 - range size 79
 - the ID map range 79
- considerations for GPFS applications 881
- contact node 460
- control file permission 188
- creating
 - access control lists 614
 - file systems 414
 - filesets 408
 - quota reports 179
- ctime 827, 830, 833, 836, 881

D

- daRebuildFailed callback 233
- data replica 373
- data replication
 - changing 35
- dataDiskWaitTimeForRecovery attribute 337
- deactivating quota limit checking 179

- deadlockBreakupDelay attribute 338
- deadlockDataCollectionDailyLimit attribute 338
- deadlockDataCollectionMinInterval attribute 338
- deadlockDetectionThreshold attribute 338
- deadlockDetectionThresholdForShortWaiters attribute 338
- deadlockDetectionThresholdIfOverloaded attribute 338
- deadlockOverloadThreshold attribute 338
- clone 738
- declustered array stanza 4
- default ACL 188
- default quotas 170
 - activating 440
 - deactivating 437
 - editing 434
- defaultHelperNodes attribute 338
- defaultMountDir attribute 338
- deleting
 - a GPFS cluster 9
 - disks 449
 - file systems 29, 458
 - filesets 455
 - nodes from a cluster 460
 - nodes from a GPFS cluster 9
 - snapshots 467
- deleting links
 - snapshots 674
- deleting, Network Shared Disks (NSDs) 465
- deny-write open lock 371
- Direct I/O caching policy 35
- direct I/O considerations 883
- directives
 - subroutine for passing 744
- directory entry
 - reading 802, 804
- directory server 73
- DirInherit 192
- disableInodeUpdateOnFdatasync attribute 339
- disabling
 - Persistent Reserve 166
- disaster recovery 496
- disk access
 - path discovery 579
- disk availability 162
- disk descriptor 355
- disk descriptors 158, 160
- disk discovery 166
- disk parameter
 - changing 354
- disk state
 - changing 163, 354
 - displaying 162
 - suspended 354
- disk status 162
- disk storage
 - pre-allocating 839
- disk usage 354, 648
- disks
 - adding 158, 239, 648
 - availability 162
 - configuration 528
 - deleting 159, 449
 - displaying information 157
 - displaying state 528
 - ENOSPC 162
 - failure 42
 - fragmentation 44
 - managing 157

- disks (*continued*)
 - maximum number 157
 - reducing fragmentation 443
 - replacing 160, 161, 648
 - status 162
 - strict replication 162
- displaying
 - access control lists 189, 500
 - cluster configuration attributes 526
 - disk fragmentation 44
 - disk state 528
 - disk states 162
 - disks 157
 - filesets 532
 - GPFS cluster configuration information 524
 - NSD belonging to a GPFS cluster 549
 - quotas 177, 559
 - snapshots 563
- displaying Quality of Service for I/O operations (QoS)
 - settings 556
- DMAPI 374
- dmapiDataEventRetry attribute 339
- dmapiEventTimeout attribute 339
- dmapiMountEvent attribute 339
- dmapiMountTimeout attribute 340
- dmapiSessionFailureTimeout attribute 340
- dynamic validation of descriptors on disk 32

E

- editing
 - default quotas 434
- enableIPv6 attribute 340
- enabling
 - Persistent Reserve 166
- enforceFilesetQuotaOnRoot attribute 340
- establishing quotas 173
- exceptions
 - NFS V4 Linux 883
- exceptions and limitations
 - GPFS applications considerations 882
- exceptions to Open Group technical standards
 - GPFS applications considerations 881
- execute file permission 187
- expelDataCollectionDailyLimit attribute 341
- expelDataCollectionMinInterval attribute 341
- expired tokens
 - deleting 105
- exporting a GPFS file system 195
- extended ACLs
 - retrieve 747
 - set 749, 751, 797
- extended attributes 519
 - Linux 882
- extended file attributes 749, 751, 797
 - retrieve 747
 - set 749, 751, 797
- external Keystone server 100

F

- failure group 354, 648
- failureDetectionTime attribute 341
- fastestPolicyCmpThreshold attribute 341
- fastestPolicyMaxValidPeriod attribute 341
- fastestPolicyMinDiffPercent attribute 341

- fastestPolicyNumReadSamples attribute 341
- file
 - access control information 769, 773, 841
 - ACL information 769, 773, 841
 - block level incremental read 810
 - extended attributes 747, 749, 751, 797
 - reading 800
- File
 - Compression 35
- file access 77
- file access pattern information 744
- file attribute
 - extended 785, 787
 - querying 519
- file attributes
 - appendOnly 321, 519
- file clone
 - copy 400, 727
 - decloning 738
 - redirect 400
 - show 400
 - snap 400, 729
 - split 400, 731
 - unsnap 733
- file descriptor
 - closing 782
 - opening 793, 795
- file permissions
 - control 188
 - GPFS extension 187
- file replication
 - querying 34
- file status information 756, 852, 854, 856
 - gpfs_stat_inode_with_xattrs() 858
 - gpfs_stat_inode_with_xattrs64() 860
- file system configuration information, backing up
 - mmbackupconfig command 52
 - using the mmbackupconfig command 52
- file system descriptor quorum 497
- file system determination
 - GPFS applications considerations 882
- file system manager
 - changing nodes 24, 379
 - displaying current 542
 - displaying node currently assigned 24
 - displaying nodes 24
- file system name 758, 761
- file system snapshot handle 754
- file system snapshots
 - subset restore 58
- file system space
 - querying 470
- file systems
 - access control lists 187
 - adding disks 239
 - AIX export 198
 - attributes
 - changing 34
 - displaying 33
 - backing up 46, 49, 281
 - block size 414
 - change manager node 379
 - changing attributes 371
 - changing attributes for files 321
 - checking 30, 487, 513
 - control request 496
 - controlled by GPFS 882
- file systems (*continued*)
 - creating 414
 - creating snapshot 431
 - deleting 458
 - deleting disks 449
 - disk fragmentation 44
 - displaying attributes 536
 - displaying format version 536
 - exporting 196, 485
 - exporting using NFS 195
 - file system manager
 - displaying 542
 - format changes 885
 - format version 371, 885
 - formatting 415
 - fragmentation
 - querying 45
 - GPFS control 882
 - handle 754
 - importing 513
 - inconsistencies 487
 - links to snapshots 674
 - Linux export 196
 - listing mounted 544
 - migrating 371, 566
 - mounted file system sizes 342, 418
 - mounting 568
 - mounting on multiple nodes 28
 - moving to another cluster 371, 485
 - mtime value 372
 - NFS export 198
 - NFS V4 export 198
 - querying space 470
 - quotas 440
 - rebalancing 642
 - reducing fragmentation 45, 443
 - remote 276, 621, 624
 - repairing 30, 487
 - restoring configuration information 631
 - restoring with snapshots 635
 - restripe 242
 - restripping 42, 642
 - space, querying 43
 - unmounting 661, 684
 - unmounting on multiple nodes 29
- fileHeatLossPercent attribute 341
- fileHeatPeriodMinutes attribute 341
- FileInherit 192
- files
 - orphaned 487
 - rebalancing 639
 - restripping 639
- files, stanza 4
- fileset quota 435, 627
- fileset snapshots
 - subset restore 59
- filesets
 - backing up 46, 47
 - changing attributes 365
 - creating 408
 - deleting 455
 - displaying 532
 - ID 789
 - linking 517
 - name 789
 - quotas 169
 - restoring with snapshots 635

- filesets (*continued*)
 - unlinking 687
- FIPS1402mode attribute 341
- FlashCopy image 496
- forceLogWriteOnFdatasync attribute 341
- FPO license 377
- full backup 816, 818, 827, 830, 833, 836

G

- gathering data to solve GPFS problems 221
- genkey 277
- GPFS 1, 8, 9
 - adminMode attribute 2
 - CES packages
 - deploying 65
 - command principles 3, 4
 - configuring cluster 7, 65
 - managing cluster 7
 - mmprotocoltrace command 607
 - node quorum 23
 - programming interfaces 735, 736, 738, 740, 742, 744, 747, 749, 751, 753, 754, 755, 756, 758, 759, 761, 763, 765, 767, 769, 771, 773, 775, 778, 782, 784, 785, 787, 789, 791, 793, 795, 797, 800, 802, 804, 806, 808, 810, 813, 814, 815, 816, 818, 820, 822, 824, 826, 827, 830, 833, 836, 839, 841, 843, 846, 848, 850, 852, 854, 856, 858, 860, 862, 863, 864, 866, 868, 869, 870, 871, 873, 875
 - stopping 661
- GPFS cache 198
- GPFS cluster
 - adding nodes 8
 - changing the GPFS cluster configuration servers 10
 - creating 7, 403
 - deleting 9
 - deleting nodes 9
 - displaying configuration information 7
 - managing 7
- GPFS cluster configuration data 485
- GPFS cluster configuration information
 - displaying 524
- GPFS cluster configuration server
 - changing 327
 - primary 328
 - secondary 328
- GPFS cluster configuration servers
 - changing 10
 - choosing 403
 - displaying 7
- GPFS cluster data 426
- GPFS commands 217
- GPFS configuration data 878, 880
- GPFS connector for Hadoop distributions
 - mmhadoopctl 506
- GPFS daemon
 - starting 25, 678
 - stopping 26, 661
- GPFS daemon status 503
- GPFS directory entry 740, 742
- GPFS file system
 - administering 1
 - adminMode attribute 2
- GPFS file system snapshot handle 758, 759, 761, 763, 765, 767, 771
 - free 753
- GPFS policy engine
 - using 52

- GPFS portability layer 292
- GPFS programming interfaces 723
- GPFS subroutines 723
- GPFS user exits 877
- gpfs_acl_t 726
- GPFS_ATTRFLAG_DEFAULT
 - gpfs_fgetattrs() 747
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
- GPFS_ATTRFLAG_FINALIZE_ATTRS
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_IGNORE_PLACEMENT
 - gpfs_igetattrsx() 787
- GPFS_ATTRFLAG_IGNORE_POOL
 - GPFS_ATTRFLAG_FINALIZE_ATTRS
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_INCL_DMAPI
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_INCL_ENCR
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_MODIFY_CLONEPARENT
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_SKIP_CLONE
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_SKIP_IMMUTABLE
 - gpfs_fgetattrs() 747
 - GPFS_ATTRFLAG_USE_POLICY
 - gpfs_fgetattrs() 747
- GPFS_ATTRFLAG_INCL_DMAPI
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_INCL_DMAPI
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_INCL_ENCR
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_MODIFY_CLONEPARENT
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_NO_PLACEMENT
 - gpfs_fgetattrs() 747
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_SKIP_CLONE
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797
- GPFS_ATTRFLAG_SKIP_IMMUTABLE
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_igetattrsx() 787
 - gpfs_iputattrsx() 797

GPFS_ATTRFLAG_USE_POLICY
 gpfs_fputattrs() 749
 gpfs_fputattrswithpathname() 751
 gpfs_igetattrsx() 787
 gpfs_iputattrsx() 797
 gpfs_clone_copy() 727
 gpfs_clone_snap() 729
 gpfs_clone_split() 731
 gpfs_clone_unsnap() 733
 gpfs_close_inodescan() 735
 gpfs_cmp_fssnapid() 736
 gpfs_declone() 738
 gpfs_direntx_t 740
 gpfs_direntx64_t 742
 gpfs_fcntl() 744
 gpfs_fgetattrs() 747
 GPFS_ATTRFLAG_DEFAULT 747
 GPFS_ATTRFLAG_FINALIZE_ATTRS 747
 GPFS_ATTRFLAG_IGNORE_POOL 747
 GPFS_ATTRFLAG_INCL_DMAPI 747
 GPFS_ATTRFLAG_INCL_ENCR 747
 GPFS_ATTRFLAG_MODIFY_CLONEPARENT 747
 GPFS_ATTRFLAG_NO_PLACEMENT 747
 GPFS_ATTRFLAG_SKIP_CLONE 747
 GPFS_ATTRFLAG_SKIP_IMMUTABLE 747
 GPFS_ATTRFLAG_USE_POLICY 747
 gpfs_fputattrs() 749
 GPFS_ATTRFLAG_DEFAULT 749
 GPFS_ATTRFLAG_FINALIZE_ATTRS 749
 GPFS_ATTRFLAG_IGNORE_POOL 749
 GPFS_ATTRFLAG_INCL_DMAPI 749
 GPFS_ATTRFLAG_INCL_ENCR 749
 GPFS_ATTRFLAG_MODIFY_CLONEPARENT 749
 GPFS_ATTRFLAG_NO_PLACEMENT 749
 GPFS_ATTRFLAG_SKIP_CLONE 749
 GPFS_ATTRFLAG_SKIP_IMMUTABLE 749
 GPFS_ATTRFLAG_USE_POLICY 749
 gpfs_fputattrswithpathname() 751
 GPFS_ATTRFLAG_DEFAULT 751
 GPFS_ATTRFLAG_FINALIZE_ATTRS 751
 GPFS_ATTRFLAG_IGNORE_POOL 751
 GPFS_ATTRFLAG_INCL_DMAPI 751
 GPFS_ATTRFLAG_INCL_ENCR 751
 GPFS_ATTRFLAG_MODIFY_CLONEPARENT 751
 GPFS_ATTRFLAG_NO_PLACEMENT 751
 GPFS_ATTRFLAG_SKIP_CLONE 751
 GPFS_ATTRFLAG_SKIP_IMMUTABLE 751
 GPFS_ATTRFLAG_USE_POLICY 751
 gpfs_free_fssnaphandle() 753
 gpfs_fssnap_handle_t 754
 gpfs_fssnap_id_t 755
 gpfs_fstat() 756
 gpfs_get_fsname_from_fssnaphandle() 758
 gpfs_get_fssnaphandle_by_fssnapid() 759
 gpfs_get_fssnaphandle_by_name() 761
 gpfs_get_fssnaphandle_by_path() 763
 gpfs_get_fssnapid_from_fssnaphandle() 765
 gpfs_get_pathname_from_fssnaphandle() 767
 gpfs_get_snapdirname() 769
 gpfs_get_snapname_from_fssnaphandle() 771
 gpfs_getacl() 773
 gpfs_iattr_t 775
 gpfs_iattr64_t 778
 gpfs_iclose() 53, 782
 gpfs_ifile_t 784
 gpfs_igetattrs() 785
 gpfs_igetattrsx() 787
 gpfs_igetattrsx() (continued)
 GPFS_ATTRFLAG_FINALIZE_ATTRS 787
 GPFS_ATTRFLAG_IGNORE_PLACEMENT 787
 GPFS_ATTRFLAG_INCL_DMAPI 787
 GPFS_ATTRFLAG_INCL_ENCR 787
 GPFS_ATTRFLAG_MODIFY_CLONEPARENT 787
 GPFS_ATTRFLAG_NO_PLACEMENT 787
 GPFS_ATTRFLAG_SKIP_CLONE 787
 GPFS_ATTRFLAG_SKIP_IMMUTABLE 787
 GPFS_ATTRFLAG_USE_POLICY 787
 gpfs_igetfilesetname() 789
 gpfs_igetstoragepool() 791
 gpfs_iopen() 53, 793
 gpfs_iopen64() 53, 795
 gpfs_iputattrsx() 797
 GPFS_ATTRFLAG_FINALIZE_ATTRS 797
 GPFS_ATTRFLAG_IGNORE_POOL 797
 GPFS_ATTRFLAG_INCL_DMAPI 797
 GPFS_ATTRFLAG_INCL_ENCR 797
 GPFS_ATTRFLAG_MODIFY_CLONEPARENT 797
 GPFS_ATTRFLAG_NO_PLACEMENT 797
 GPFS_ATTRFLAG_SKIP_CLONE 797
 GPFS_ATTRFLAG_SKIP_IMMUTABLE 797
 GPFS_ATTRFLAG_USE_POLICY 797
 gpfs_iread() 53, 800
 gpfs_ireaddir() 53, 802
 gpfs_ireaddir64() 53, 804
 gpfs_ireadlink() 806
 gpfs_ireadlink64() 808
 gpfs_ireadx() 810
 gpfs_iscan_t 813
 gpfs_lib_init() 814
 gpfs_lib_term() 815
 gpfs_next_inode_with_xattrs() 820
 gpfs_next_inode_with_xattrs64() 822
 gpfs_next_inode() 53, 816
 gpfs_next_inode64() 53, 818
 gpfs_next_xattr() 824
 gpfs_opaque_acl_t 826
 gpfs_open_inodescan_with_xattrs() 833
 gpfs_open_inodescan_with_xattrs64() 836
 gpfs_open_inodescan() 53, 827
 gpfs_open_inodescan64() 53, 830
 gpfs_prealloc() 839
 gpfs_putacl() 841
 gpfs_quotactl() 170, 843
 gpfs_quotaInfo_t 846
 gpfs_seek_inode() 848
 gpfs_seek_inode64() 850
 gpfs_stat_inode_with_xattrs() 858
 gpfs_stat_inode_with_xattrs64() 860
 gpfs_stat_inode() 854
 gpfs_stat_inode64() 856
 gpfs_stat() 852
 GPFS-specific
 mount options 28
 gpfs.snap command 221
 gpfsFcntlHeader_t 862
 gpfsGetFilesetName_t 863
 gpfsGetReplication_t 864
 gpfsGetSetXAttr_t 866
 gpfsGetSnapshotName_t 868
 gpfsGetStoragePool_t 869
 gpfsListXAttr_t 870
 gpfsRestripeData_t 871
 gpfsSetReplication_t 873
 gpfsSetStoragePool_t 875

- grace period
 - changing 481, 657
 - setting 481, 657
- group quota 435, 437, 440, 482, 559, 619, 627
- GUI 183
- GUI administrators 183

H

- Hadoop distributions, GPFS connector for
 - mmhadoopctl 506
- hints
 - subroutine for passing 744
- hole 810

I

- I/O caching policy
 - changing 321
- IBM Spectrum Scale 1, 2, 3, 4, 109, 115, 125, 135
 - access control lists 189, 193, 207
 - administration 191
 - applying 194
 - change 190, 194
 - delete 190, 195
 - display 194
 - exceptions 195
 - limitations 195
 - setting 188, 194
 - syntax 191
 - translation 193
 - access control lists (ACL)
 - best practices 202
 - inheritance 201
 - permissions 203
 - ACL administration 187
 - activating quota limit checking 178
 - active connections to SMB export 118
 - Add disks 158
 - adding node 8
 - administering unified file and object access 142
 - example scenario 147
 - associate containers 146
 - authorizing protocol users 200
 - CES packages
 - deploying 65
 - change GPFS disk states 163
 - change GPFS parameters 163
 - change NSD configuration 165
 - change Object configuration values 126
 - change quota limit checking 179
 - changing NFS export configuration 120
 - changing the GPFS cluster configuration data 10
 - check quota 176
 - cluster configuration information 7
 - configuring cluster 7
 - create export on container 146
 - create NFS export 120
 - create SMB share ACLs 114
 - Create SMB shares 116
 - Creating SMB share 113
 - creating storage policy 145
 - data ingestion 151
 - deactivating quota limit checking 179
 - delete disk 159
 - deleting node 9

- IBM Spectrum Scale (*continued*)
 - disconnect active connections to SMB 118
 - disk availability 162
 - disk status 162
 - Disks in a GPFS cluster 157
 - display GPFS disk states 162
 - enable file-access object capability 142
 - Enable object access 146
 - establish and change quotas 173
 - export file systems 196
 - file system quota report
 - create 179
 - file systems 196
 - AIX export 198
 - GPFS access control lists (ACLs)
 - manage 187
 - GPFS cache usage 198
 - GPFS quota management
 - disable 169
 - enable 169
 - identity management modes for unified file and object access 136
 - in-place analytics 149
 - limitations of unified file and object access 150
 - Linux export 196
 - list NFS export 121
 - list quota information 177
 - list SMB shares 114
 - Manage default quotas 170
 - manage disk 157
 - manage GPFS quotas 169
 - manage GUI administrators 183
 - Manage NFS exports 120
 - Managing ACLs of SMB exports 117
 - managing cluster 7
 - Managing OpenStack ACLs 129
 - managing protocol data exports 113
 - managing SMB shares 113
 - Mapping OpenStack commands
 - administrator commands 125
 - Modifying SMB exports 116, 117
 - multi-region object deployment 134
 - addin region 132
 - multiprotocol export considerations 121
 - multiprotocol exports 121
 - Network File System (NFS) 195
 - NFS 196, 198
 - NFS automount 199
 - NFS export 198
 - Unmount a file 199
 - NFS export configuration 198, 199
 - node quorum 23
 - node quorum with tiebreaker 23
 - NSD server
 - Change server usage and failback 166
 - objectizer 140
 - Persistent Reserve (PR) functionality
 - disable 166
 - enable 166
 - programming interfaces 735, 736, 738, 740, 742, 744, 747, 749, 751, 753, 754, 755, 756, 758, 759, 761, 763, 765, 767, 769, 771, 773, 775, 778, 782, 784, 785, 787, 789, 791, 793, 795, 797, 800, 802, 804, 806, 808, 810, 813, 814, 815, 816, 818, 820, 822, 824, 826, 827, 830, 833, 836, 839, 841, 843, 846, 848, 850, 852, 854, 856, 858, 860, 862, 863, 864, 866, 868, 869, 870, 871, 873, 875

- IBM Spectrum Scale *(continued)*
 - quotas
 - NFS 172
 - SMB 172
 - remote login 22
 - Remove NFS export 121
 - remove SMB shares 114
 - replace disk 160
 - restore quota files 180
 - security mode 20
 - set quota 174
 - set up objectizer service interval 143
 - SMB and NFS protocols 121
 - SMB share configuration 114
 - SMB share limitations 119
 - storage policies for objects 130
 - strict disk replication 162
 - sudo wrapper 21
 - sudo wrapper scripts 22
 - synchronous write operations 199
 - unified file and object access 135, 140, 141, 149
 - unified file and object access constraints 151
 - unified file and object access modes 135
 - using storage policy 145
 - view file locks in SMB export 119
 - view open files in SMP export 118
- IBM Spectrum Scale cluster
 - creating 7
- IBM Spectrum Scale file attributes
 - modify 34
- IBM Spectrum Scale file system
 - checking 30
 - repairing 30
- IBM Spectrum Scale file system attributes 33
- IBM Spectrum Scale file systems
 - deleting 29
 - management 27
 - mount options 28
 - mounting 27, 28
 - which nodes have mounted 30
- IBM Spectrum Scale for object storage
 - administering storage policies 131
 - authentication
 - configuring 93
 - configuration files 153
 - create accounts 104
 - managing 123
 - managing object capabilities 130
 - S3 API emulation 126
 - services 123
 - storage policies to fileset mapping 130
 - storage policy for compression 132
 - unified file and object access related user tasks 152
- IBM Spectrum Scale information units xi
- IBM Spectrum Scale unmounting a file system 29
- IBM Spectrum Scale user exits 877
- IBM Spectrum Scale for object storage
 - EC2 credentials 128
- IBM Spectrum Scale IBM Spectrum Scale
 - configure authentication 143
 - set identity management modes 143
- ibmobjectizer service 140
- identity management mode for unified file and object access
 - local_mode 135
- identity management modes unified file and object access
 - unified_mode 136
- image backup 53
 - image restore 53
 - in-doubt value 361, 560, 628
 - in-place analytics 149
 - incremental backup 816, 818, 827, 830, 833, 836
 - inheritance flags 201
 - inheritance of ACLs 191
 - DirInherit 192
 - FileInherit 192
 - Inherited 192
 - InheritOnly 192
 - Inherited 192
 - InheritOnly 192
 - inode
 - attributes 775, 778
 - inode file handle 782, 784
 - inode number 793, 795, 806, 808, 816, 818, 820, 822, 824, 827, 830, 833, 836, 848, 850
 - inode scan 810, 816, 818, 820, 822, 824, 848, 850
 - closing 735
 - opening 827, 830, 833, 836
 - inode scan handle 735, 813
 - installation 711
 - interface calls, cleanup after 815
 - interface for additional calls, setup of 814
 - iscan handle 735

K

- kernel memory 321
- Keystone
 - expired tokens 105
- Keystone tokens
 - deleting 105

L

- LDAP
 - bind user requirements 71
 - LDAP server 71
 - LDAP user information 74
 - LDAP-based authentication for file access 84
 - LDAP with Kerberos 86
 - LDAP with TLS 85
 - LDAP with TLS and Kerberos 87
 - LDAP without TLS and Kerberos 88
 - LDAP-based authentication for object access 98
 - LDAP with TLS 99
 - LDAP without TLS 99
- license 377
- limitations
 - NFS V4 Linux 883
- Limitations
 - of the mmuserauth service create command 83
- linking
 - filesets 517
- links to snapshots
 - creating 674
 - deleting 674
- listing
 - snapshots 563
 - user-defined callbacks 522
- listing Quality of Service for I/O operations (QoS)
 - settings 556
- local authentication for object access 94, 95
- local snapshots
 - subset restore 58, 59

- local snapshots *(continued)*
 - subset restore using script 60
- lost+found directory 31
- lrocData attribute 342
- lrocDataMaxFileSize attribute 342
- lrocDataStubFileSize attribute 342
- lrocDirectories attribute 342
- lrocInodes attribute 342

M

- managing
 - a GPFS cluster 7
 - GPFS quotas 169
 - GUI administrators 183
- Managing
 - protocol services 66
- maxblocksize attribute 342
- maxDownDisksForRecovery attribute 342
- maxFailedNodesForRecovery attribute 343
- maxFcntlRangesPerFile attribute 343
- maxFilesToCache attribute 343
- maximum number of files
 - changing 371
 - displaying 536
- maxMBpS attribute 343
- maxStatCache attribute 343
- metadata 322
- metadata replica 372
- metadata replication
 - changing 35
- metadataDiskWaitTimeForRecovery attribute 343
- minDiskWaitTimeForRecovery attribute 343
- mmaddcallback 226
- mmadddisk 158, 239, 880
- mmaddnode 8, 245
- mmafmconfig 248
- mmafmctl 251
- mmafmlocal 264
- mmapplypolicy 46, 266
- mmapRangeLock attribute 344
- mmauth 20, 276
- mmbackup 46, 47, 48, 49, 50, 281
- MMBACKUP_PROGRESS_CALLOUT 51
- mmbackupconfig 46, 290
- mmbuildgpl 292
- MMC
 - connect SMB exports 115
 - connect SMB shares 115
 - create SMB exports 116
 - create SMB shares 116
 - manage SMB export ACLs 117
 - manage SMB exports 115
 - manage SMB shares 115
 - modify SMB exports 116
 - remove SMB exports 116
 - SMB export active connections 118
 - SMB export disconnect connections 118
 - SMB export offline settings 117
 - SMB export open files 118
 - SMB export view file locks 119
- mmcallhome 293
- mmces 304
- mmcesdr 313
- mmchattr 34, 35, 42, 321
- mmchcluster 10, 327
- mmchconfig 11, 20, 197, 198, 331

- mmchdisk 42, 162, 163, 354
- mmcheckquota 169, 176, 179, 361
- mmchfileset 365
- mmchfs 34, 166, 169, 178, 179, 198, 371
- mmchlicense 377
- mmchmgr 379
- mmchnode 381
- mmchnodeclass 385
- mmchnsd 388
- mmchpolicy 391
- mmchpool 394
- mmchqos 396
- mmclone 400
- mmcrcluster 7, 403
- mmcrfileset 408
- mmcrfs 27, 169, 178, 191, 198, 414
- mmcrnodeclass 424
- mmcrnsd 158, 426, 648, 879, 880
- mmcrsnapshot 53, 431
- mmdefedquota 434
- mmdefquotaoff 437
- mmdefquotaon 440
- mmdefragfs 44, 45, 443
- mmdelacl 190, 191, 195, 446
- mmdelcallback 448
- mmdeldisk 159, 449, 880
- mmdelfileset 455
- mmdelfs 29, 458
- mmdelnode 9, 460
- mmdelnodeclass 463
- mmdelnsd 465
- mmdelsnapshot 467
- mmdf 43, 159, 470
- mmdiag 473
- mmeditacl 190, 191, 193, 194, 478
- mmedquota 169, 173, 481
- mmexportfs 485
- MMFS_FSSTRUCT 487
- MMFS_SYSTEM_UNMOUNT 488
- mmfsck 159, 487
- mmfsctl 496
- mmgetacl 188, 189, 193, 194, 195, 500
- mmgetstate 503
- mmhadoopctl 506
- mmimgbackup 508
- mmimgrestore 511
- mmimportfs 513
- mmlinkfileset 517
- mmlsattr 34, 519
- mmlscallback 522
- mmlscluster 7, 524
- mmlsconfig 526
- mmlsdisk 162, 528
- mmlsfileset 532
- mmlsfs 33, 162, 178, 179, 198, 536
- mmlslicense 540
- mmlsmgr 24, 542
- mmlsmount 30, 544
- mmlsnodeclass 546
- mmlsnsd 157, 549
- mmlspolicy 552
- mmlspool 554
- mmlsqos 556
- mmlsquota 177, 559
- mmlssnapshot 563, 777, 781
- mmmigratefs 566
- mmmount 27, 28, 166, 568

- mmnfs 570
- mmnfs export add command 120
- mmnsddiscover 579
- mmobj 581
- mmobj command
 - changing Object configuration values 126
- mmperfmon query 592
- mmpmon 602
- mmprotocoltrace 607
- mmpsnap 611
- mmputacl 188, 189, 191, 194, 195, 614
- mmquotaoff 178, 179, 617
- mmquotaon 178, 619
- mmremotecluster 621
- mmremotefs 166, 624
- mmrepquota 179, 627
- mmrestoreconfig 631
- mmrestorefs 635
- mmrestripefile 639
- mmrestripefs 42, 43, 159, 162, 642
 - completion time 24
- mmrpldisk 648
- mmsdrrestore 655
- mmsetquota 169, 657
- mmshutdown 26, 661
- mmsmb 663
 - list SMB shares 114
- mmsnapdir 674, 754
- mmstartup 25, 678
- mmtracectl 680
- mmumount 29, 684
- mmunlinkfileset 687
- mmuserauth 69, 76, 83, 690
- mmwinserv service
 - managing 709
- mmwinservctl 709
- modifying file system attributes 34
- monitoring
 - performance 602
- mount point directory 415
- mounting
 - file systems 27
- mounting a file system 371
 - an NFS exported file system 195
- mtime 418, 827, 830, 833, 836, 852, 854, 856, 881
- multi-region object deployment
 - adding region 132
 - administering 134
 - exporting configuration data 134
 - importing configuration data 134
 - mmobj command 581
 - removing region 134
- Multiprotocol export considerations
 - NFS export 121
 - SMB export 121

N

- Network File System (NFS)
 - cache usage 198
 - exporting a GPFS file system 195
 - interoperability with GPFS 195
 - synchronous writes 199
 - unmounting a file system 199
- Network Information Server 89
- Network Shared Disks (NSDs) 879
 - changing configuration attributes 165, 388

- Network Shared Disks (NSDs) (*continued*)
 - creating 426
 - displaying 549
- Network Shared Disks (NSDs), deleting 465
- NFS
 - quotas 172
- NFS automount 199
- NFS export
 - create NFS export 120
 - list NFS export 121
- NFS exports
 - Manage NFS exports 120
- NFS protocol services
 - starting 67
- NFS V4 187, 371, 418
- NFS V4 ACL 372, 446, 478, 479, 500, 501, 614
 - GPFS exceptions 882
 - special names 882
- NFS V4 Linux limitations 883
- NFS V4 protocol
 - GPFS exceptions 882
- NIS-based authentication for file access 89
- nistCompliance attribute 344
- node classes, user-defined 3
 - changing 385
 - creating 424
 - deleting 463
 - listing 546
- node descriptor 245, 403
- node designation 245, 403
- node failure detection 341
- node quorum 23
- node quorum with tiebreaker 10, 23
- nodes
 - adding to a cluster 245
 - adding to a GPFS cluster 8
 - assigned as file system manager 24
 - deleting from a cluster 460
 - specifying with commands 3
 - which have file systems mounted 30
- noSpaceEventInterval attribute 344
- NSD fallback 166
- NSD path 579
- NSD server 25, 166, 513
- NSD server list
 - changing 165, 388
- NSD server nodes
 - changing 165, 388
 - choosing 426
- NSD stanza 4
- NSD volume ID 426, 465
- nsdBufSpace attribute 344
- nsdCksumMismatch callback 235
- nsdRAIDBufferPoolSizePct attribute 344
- nsdRAIDTracks attribute 344
- nsdServerWaitTimeForMount attribute 345
- nsdServerWaitTimeWindowOnMount attribute 345
- numaMemoryInterleave attribute 345

O

- object capabilities
 - disabling 130
 - enabling 130
 - listing 130
 - managing 130

- object Configuration values
 - Changing 126
- Object protocol service
 - starting 68
- object services
 - tuning 69
- object storage
 - containers 209
 - create accounts 104
 - creating containers 209
 - managing 123
- object storage authentication
 - configuring 93
- object storage services
 - managing 123
- objectization 140
- objectizer 140
- objects
 - managing
 - endpoints 101
 - projects 101
 - roles 101
 - users 101
- OpenLDAP
 - server ACLs 72
- OpenStack ACLs
 - managing 129
 - using S3 API emulation 129
- OpenStack commands
 - Mapping 125
- OpenStack EC2 credentials
 - configuring 128
- options
 - always 29
 - asfound 29
 - asneeded 29
 - atime 28
 - mtime 28
 - never 29
 - noatime 28
 - nomtime 28
 - norelatime 28
 - nosyncnfs 28
 - relatime 28
 - syncnfs 29
 - useNSDserver 29
- orphaned files 31

P

- pagepool attribute 345
- pagepoolMaxPhysMemPct attribute 346
- pdFailed callback 235
- pdPathDown callback 235
- pdRecovered callback 235
- pdReplacePdisk callback 235
- peer recovery cluster 496
- Peer-to-Peer Remote Copy (PPRC) 496
- performance tuning
 - object services 69
- performance, monitoring 602
- Persistent Reserve
 - disabling 166
 - enabling 166
- physical disk stanza 4
- policy
 - applying 266

- policy rules 35
- pool
 - displaying 554
- postRGRelinquish callback 236
- postRGTakeover callback 236
- PR 166
- prefetchThreads attribute 346
- prerequisite
 - Kerberos-based SMB access 81
- prerequisites
 - LDAP server 71
- preRGRelinquish callback 235
- preRGTakeover callback 236
- primary GPFS cluster configuration server 405
- principles
 - common to GPFS commands 3, 4
- protocols
 - administration tasks 63, 112
 - removal tasks 63
- protocols data exports 113
- public/private key pair 276

Q

- QoS Classes
 - maintenance 40
 - other 40
- Quality of Service for I/O operations (QoS)
 - configuring 40
- Quality of Service for I/O operations (QoS) level
 - changing 396
- Quality of Service for I/O operations (QoS) settings
 - listing 556
- querying
 - disk fragmentation 44
 - file system fragmentation 45
 - replication 34
 - space 43
- Querying file system 43
- quorum 497
- quorum node 403, 460, 503
- quota files
 - backing up 180
 - replacing 361
 - restoring 180
- quota information 846
- quotas
 - activating 619
 - activating limit checking 178
 - changing 173, 481, 657, 843
 - changing limit checking 179
 - checking 176, 361
 - creating reports 179, 627
 - deactivating 617
 - deactivating limit checking 179
 - default values 170
 - disabling 169
 - displaying 177, 559
 - enabling 169
 - establishing 173
 - fileset 169
 - group 169
 - setting 481, 657
 - user 169

R

- RAID stripe size 414
- read file permission 187
- readReplicaPolicy attribute 346
- rebalancing a file 639
- rebalancing a file system 642
- recovery group stanza 4
- recovery groups
 - stanza files 515
- refresh NSD server
 - mmnsddiscover 579
- registering user event commands 226
- release attribute 347
- remote copy command
 - changing 327
 - choosing 403
- remote file systems 276
- remote shell command
 - changing 327
 - choosing 403
- repairing
 - file system 30
- replacing disks 160, 161, 648
- replicated cluster 880
- replication 372
 - changing 34, 35
 - querying 34, 519
- replication attributes
 - changing 321
- replication factor 321
- replication, strict 419
- requirements
 - administering GPFS 1
 - for Tivoli Storage Manager 48
- restoring configuration information 631
- restoring from local snapshots
 - using the sample script 60
- restoring NSD path
 - mmnsddiscover 579
- restripeOnDiskFailure attribute 347
- restriping a file 639
- restriping a file system 42, 642
- rgOpenFailed callback 236
- rgPanic callback 237
- rpcPerfNumberDayIntervals attribute 347
- rpcPerfNumberHourIntervals attribute 347
- rpcPerfNumberMinuteIntervals attribute 347
- rpcPerfNumberSecondIntervals attribute 348
- rpcPerfRawExecBufferSize attribute 348
- rpcPerfRawStatBufferSize attribute 348

S

- S3 ACLs
 - managing 129
- S3 API emulation
 - enabling 126
- samba attributes 74
- Scale Out Backup and Restore 53
- secondary GPFS cluster configuration server 405
- security mode
 - managing remote access 20
- selective objectization 146
- Server license 377
- server node
 - restoring NSD path 579

- server node, NSD
 - choosing 426
- set up authentication servers 70
 - integrating with AD server 70
 - integrating with Keystone Identity Service 76
 - integrating with LDAP server 71
- Setting
 - LDAP server prerequisites 71
- setting access control lists 188
- setting quotas
 - per-project 174
- setup of interface for additional calls 814
- sidAutoMapRangeLength attribute 348
- sidAutoMapRangeStart attribute 348
- SMB
 - export 592
 - quotas 172
- SMB exports
 - active connections 118
 - connecting 115
 - creating 116
 - disconnect connections 118
 - managing 115
 - managing ACLs 117
 - modifying 116
 - offline settings 117
 - removing 116
 - view file locks 119
 - view open files 118
- SMB protocol services
 - starting 67
- SMB shares
 - active connections 118
 - connecting 115
 - creating 116
 - disconnect connections 118
 - GUI navigation 113
 - managing 115
 - managing ACLs 117
 - managing SMB shares 113
 - modifying 116
 - offline settings 117
 - removing 116
 - view file locks 119
 - view open files 118
- snapshot
 - temporary 49
- snapshot directory 769
- snapshot handle 754, 758, 759, 761, 763, 765, 767, 771
 - free 753
- snapshot ID 754, 755, 759, 765
 - comparing 736
 - internal 777, 781
- snapshot name 761, 771
- snapshots
 - creating 431
 - deleting 431, 467
 - directory 431
 - displaying 563
 - fileset 431
 - global 431
 - listing 563
 - restoring a file system 635
 - restoring a fileset 635
- SOBAR 53
- sort-command parameter of mmappypolicy command 273
- sparse file 810

- spectrumscale 112, 711
- standards, exceptions to 881
- stanza files 4
 - recovery group 515
- stanza, declustered array 4
- stanza, NSD 4
- stanza, physical disk 4
- stanza, recovery group 4
- stanza, virtual disk 4
- starting GPFS 25, 678
 - before starting 25
- status
 - disk 162
- stopping GPFS 25, 26
- storage
 - pre-allocating 839
 - storage policies 130, 132
 - storage policies for object
 - administering 131
 - compression 132
 - mapping to filesets 130
 - storage pool 554
 - storage pool properties
 - changing 394
 - storage pools
 - ID 791
 - name 791
 - strict replication 372, 419
 - structures
 - gpfs_acl_t 726
 - gpfs_direntx_t 740
 - gpfs_direntx64_t 742
 - gpfs_fssnap_handle_t 754
 - gpfs_fssnap_id_t 755
 - gpfs_iattr_t 775
 - gpfs_iattr64_t 778
 - gpfs_ifile_t 784, 785, 787
 - gpfs_iscan_t 813
 - gpfs_opaque_acl_t 826
 - gpfs_quotaInfo_t 846
 - gpfsFcntlHeader_t 862
 - gpfsGetFilesetName_t 863
 - gpfsGetReplication_t 864
 - gpfsGetSetXAttr_t 866
 - gpfsGetSnapshotName_t 868
 - gpfsGetStoragePool_t 869
 - gpfsListXAttr_t 870
 - gpfsRestripeData_t 871
 - gpfsSetReplication_t 873
 - gpfsSetStoragePool_t 875
- subnets attribute 348
- subroutine
 - gpfs_close_inodescan() 735
 - gpfs_cmp_fssnapid() 736
- subroutines
 - gpfs_clone_copy() 727
 - gpfs_clone_snap() 729
 - gpfs_clone_split() 731
 - gpfs_clone_unsnap() 733
 - gpfs_declone() 738
 - gpfs_fcntl() 744
 - gpfs_fgetattrs() 747
 - gpfs_fputattrs() 749
 - gpfs_fputattrswithpathname() 751
 - gpfs_free_fssnaphandle() 753
 - gpfs_fstat() 756
 - gpfs_get_fsname_from_fssnaphandle() 758
- subroutines (*continued*)
 - gpfs_get_fssnaphandle_by_fssnapid() 759
 - gpfs_get_fssnaphandle_by_name() 761
 - gpfs_get_fssnaphandle_by_path() 763
 - gpfs_get_fssnapid_from_fssnaphandle() 765
 - gpfs_get_pathname_from_fssnaphandle() 767
 - gpfs_get_snapdirname() 769
 - gpfs_get_snapname_from_fssnaphandle() 771
 - gpfs_getacl() 773
 - gpfs_iclose() 53, 782
 - gpfs_igetattrs() 785
 - gpfs_igetattrsx() 787
 - gpfs_igetfilesetname() 789
 - gpfs_igetstoragepool() 791
 - gpfs_iopen() 53, 793
 - gpfs_iopen64() 53, 795
 - gpfs_iputattrsx() 797
 - gpfs_iread() 53, 800
 - gpfs_ireaddir() 53, 802
 - gpfs_ireaddir64() 53, 804
 - gpfs_ireadlink() 806
 - gpfs_ireadlink64() 808
 - gpfs_ireadx() 810
 - gpfs_lib_init() 814
 - gpfs_lib_term() 815
 - gpfs_next_inode_with_xattrs() 820
 - gpfs_next_inode_with_xattrs64() 822
 - gpfs_next_inode() 53, 816
 - gpfs_next_inode64() 53, 818
 - gpfs_next_xattr() 824
 - gpfs_open_inodescan_with_xattrs() 833
 - gpfs_open_inodescan_with_xattrs64() 836
 - gpfs_open_inodescan() 53, 827
 - gpfs_open_inodescan64() 53, 830
 - gpfs_prealloc() 839
 - gpfs_putacl() 841
 - gpfs_quotactl() 170, 843
 - gpfs_seek_inode() 848
 - gpfs_seek_inode64() 850
 - gpfs_stat_inode_with_xattrs() 858
 - gpfs_stat_inode_with_xattrs64() 860
 - gpfs_stat_inode() 854
 - gpfs_stat_inode64() 856
 - gpfs_stat() 852
- sudo wrapper 21
- sudo wrapper scripts
 - configuring on existing cluster 22
 - configuring on new cluster 22
- swift workers
 - tuning 69
- symbolic link
 - reading 806, 808
- syncFSconfig 496
- system snapshots 221
- systemLogLevel attribute 349

T

- temporary snapshot
 - backing up to the TSM server 49
- tiebreakerDisks attribute 349
- timeout period 661
- tivoli directory server
 - ACLs 73
- Tivoli Storage Manage 56
 - for IBM Spectrum Scale 56
- Tivoli Storage Manager 54

- Tivoli Storage Manager (*continued*)
 - backup scheduler 54
 - configuration specifics 54, 57
 - dsm.opt 56
 - dsm.sys 54
 - for IBM Spectrum Scale 54
 - scheduling backups 54
- Tivoli Storage Manager (TSM)
 - using the mmbackup command 281
- Tivoli Storage Manager backup planning 54, 57
 - dsm.opt options 56
 - dsm.sys options 54
- Tivoli Storage Manager backup scheduler 54
- Tivoli Storage Manager requirements 48
- trace-recycle
 - changing 681
- tracing attributes, changing 680
- traditional ACL 372, 478, 479, 500, 501
- traditional ACLs
 - NFS V4 ACL 418
 - Windows 418
- TSM interface 49

U

- UID domain 405
- uidDomain attribute 349
- unified file and object access
 - administering 135, 142
 - associating container 146
 - authentication 140
 - configuration files 153
 - configuring authentication 143
 - constraints 151
 - creating NFS export 146
 - creating SMB export 146
 - creating storage policy 145
 - data ingestion through object 151
 - example scenario 147
 - examples 151
 - file path 141
 - determining 141
 - file-access capability 142
 - identity management modes 135, 143
 - limitations 150
 - managing 135
 - mmobj command 581
 - object path 141
 - determining 141
 - object-server-sof.conf 153
 - objectization 140
 - objectizer service interval 143
 - POSIX path 141
 - determining 141
 - scheduling objectizer 143
 - selective objectization 146
 - setting up mode 143
 - spectrum-scale-object.conf 153
 - spectrum-scale-objectizer.conf 153
 - unified_mode identity management 136
 - use cases 149
- unified file and object access modes 135
- unified file and object access related user tasks
 - curl commands 152
- unified file and object access storage policy 145
 - associate container 146
 - creating export 146

- unlinking
 - filesets 687
- unmounting a file system 29
 - NFS exported 199
 - on multiple nodes 29
- unmountOnDiskFail attribute 349
- update 74
- useNSDserver
 - values 29
- usePersistentReserve attribute 350
- user event commands, registering 226
- user exit
 - GPFS 880
 - IBM Spectrum Scale 880
- user exits 877
 - GPFS 879
 - IBM Spectrum Scale 879
 - mmsdrbackup 878
 - nsdddevices 879
 - syncfsconfig 880
- user quota 435, 437, 440, 482, 559, 619, 627
- user space buffer 321
- user-defined callbacks
 - deleting 448
 - listing 522
- user-defined node classes 3
 - changing 385
 - creating 424
 - deleting 463
 - listing 546
- using a clustered NFS subsystem 199
- using the GPFS policy engine 52
- using the gpfs.snap command
 - gathering data 221

V

- validation of descriptors on disk dynamically 32
- verbsPorts attribute 350
- verbsRdma attribute 351
- verbsRdmaCm attribute 351
- verbsRdmaRoCEToS attribute 351
- verbsRdmaSend attribute 351
- verbsRdmPerConnection attribute 351
- verbsRdmPerNode attribute 351
- verbsSendBufferMemoryMB attribute 351
- virtual disk stanza 4

W

- worker1Threads attribute 352
- write file permission 187



Product Number: 5725-Q01
5641-GPF
5725-S28

Printed in USA

SA23-1452-06

