IBM Spectrum Scale
Version 4 Release 2.1

Command and Programming Reference

SA23-1456-01

IBM Spectrum Scale
Version 4 Release 2.1

Command and Programming Reference

SA23-1456-01

Note
Before using this information and the product it supports, read the information in "Notices" on page 767.

This edition applies to version 4 release 2 modification 1 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 xii. 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 vii

About this information ix
Prerequisite and related information xi
Conventions used in this information xi
How to send your comments xii

Summary of changes xiii

Chapter 1. Command reference. 1
gpfs.snap command 5
mmaddcallback command 10
mmadddisk command 23
mmaddnode command 29
mmadquery command. 32
mmafmconfig command 37
mmafmctl command 40
mmafmlocal command 54
mmapplypolicy command 56
mmauth command 67
mmbackup command 72
mmbackupconfig command 81
mmbuildgpl command 83
mmcallhome command 85
mmces command 96
mmcesdr command 105
mmchattr command 113
mmchcluster command 119
mmchconfig command 123
mmchdisk command 151
mmcheckquota command 159
mmchfileset command 163
mmchfs command. 169
mmchlicense command 175
mmchmgr command 178
mmchnode command 180
mmchnodeclass command 185
mmchnsd command 188
mmchpolicy command 191
mmchpool command. 194
mmchqos command 196
mmclone command 201
mmcloudgateway command 204
mmcrcluster command 218
mmcrfileset command 223
mmcrfs command 229
mmcrnodeclass command 239
mmcrnsd command 241
mmcrsnapshot command 246
mmdefedquota command 251
mmdefquotaoff command 254
mmdefquotaon command 257
mmdefragfs command 260
mmdelacl command 263
mmdelcallback command 265
mmdeldisk command 266
mmdelfileset command 271
mmdelfs command 274
mmdelnode command 276
mmdelnodeclass command 279
mmdelnsd command 281
mmdelsnapshot command 283
mmdf command 287
mmdiag command 290
mmeditacl command 295
mmedquota command 298
mmexportfs command 302
mmfsck command. 304
mmfsctl command. 313
mmgetacl command 317
mmgetstate command 320
mmhadoopctl command. 323
mmhealth command 324
mmimgbackup command 329
mmimgrestore command 333
mmimportfs command 336
mmkeyserv command 340
mmlinkfileset command 350
mmlsattr command 352
mmlscallback command 355
mmlscluster command 357
mmlsconfig command 359
mmlsdisk command 361
mmlsfileset command 365
mmlsfs command 369
mmlslicense command 373
mmlsmgr command 375
mmlsmount command 377
mmlsnodeclass command 379
mmlsnsd command 381
mmlspolicy command 384
mmlspool command 386
mmlsqos command 388
mmlsquota command 391
mmlssnapshot command 395
mmmigratefs command 398
mmmount command 400
mmnfs command 402
mmnsddiscover command 412
mmobj command 414
mmperfmon command 428
mmpmon command 438
mmprotocoltrace command. 443
mmpsnap command 447
mmputacl command 450
mmquotaoff command 453
mmquotaon command 455
mmremotecluster command 457
mmremotefs command 460
mmrepquota command 463
mmrestoreconfig command. 467

© Copyright IBM Corp. 2014, 2016

mmrestorefs command 471
mmrestripefile command 475
mmrestripefs command 478
mmrpldisk command. 485
mmsdrrestore command. 492
mmsetquota command 494
mmshutdown command. 498
mmsmb command. 500
mmsnapdir command 511
mmstartup command. 515
mmtracectl command. 517
mmumount command 521
mmunlinkfileset command 524
mmuserauth command 527
mmwinservctl command 546
spectrumscale command. 548

Chapter 2. IBM Spectrum Scale Data Management API for GPFS information 565
Overview of IBM Spectrum Scale Data Management API for GPFS 565
GPFS-specific DMAPI events 565
DMAPI functions 566
DMAPI configuration attributes 570
DMAPI restrictions for GPFS 571
Concepts of IBM Spectrum Scale Data Management API for GPFS 572
Sessions 572
Data management events 572
Mount and unmount 574
Tokens and access rights 575
Parallelism in Data Management applications 576
Data Management attributes 577
Support for NFS 577
Quota 577
Memory mapped files 577
Administration of IBM Spectrum Scale Data Management API for GPFS 578
Required files for implementation of Data Management applications 578
GPFS configuration attributes for DMAPI . . . 579
Enabling DMAPI for a file system 580
Initializing the Data Management application 581
Specifications of enhancements for IBM Spectrum Scale Data Management API for GPFS 581
Enhancements to data structures 581
Usage restrictions on DMAPI functions. . . . 583
Definitions for GPFS-specific DMAPI functions 584
Semantic changes to DMAPI functions 598
GPFS-specific DMAPI events 599
Additional error codes returned by DMAPI functions 600
Failure and recovery of IBM Spectrum Scale Data Management API for GPFS 601
Single-node failure 602
Session failure and recovery 603
Event recovery 603
Loss of access rights 604
DODeferred deletions 604
DM application failure 604

Chapter 3. GPFS programming interfaces 607
gpfs_acl_t structure 610
gpfs_clone_copy() subroutine 611
gpfs_clone_snap() subroutine 613
gpfs_clone_split() subroutine 615
gpfs_clone_unsnap() subroutine 617
gpfs_close_inodescan() subroutine 619
gpfs_cmp_fssnapid() subroutine 620
gpfs_declone() subroutine 622
gpfs_direntx_t structure 624
gpfs_direntx64_t structure 626
gpfs_fcntl() subroutine 628
gpfs_fgetattrs() subroutine 631
gpfs_fputattrs() subroutine 633
gpfs_fputattrswithpathname() subroutine 635
gpfs_free_fssnaphandle() subroutine. 637
gpfs_fssnap_handle_t structure 638
gpfs_fssnap_id_t structure 639
gpfs_fstat() subroutine 640
gpfs_get_fsname_from_fssnaphandle() subroutine 642
gpfs_get_fssnaphandle_by_fssnapid() subroutine 643
gpfs_get_fssnaphandle_by_name() subroutine . . 645
gpfs_get_fssnaphandle_by_path() subroutine . . . 647
gpfs_get_fssnapid_from_fssnaphandle() subroutine 649
gpfs_get_pathname_from_fssnaphandle() subroutine 651
gpfs_get_snapdirname() subroutine 653
gpfs_get_snapname_from_fssnaphandle() subroutine 655
gpfs_getacl() subroutine 657
gpfs_iattr_t structure 659
gpfs_iattr64_t structure 662
gpfs_iclose() subroutine 666
gpfs_ifile_t structure 668
gpfs_igetattrs() subroutine 669
gpfs_igetattrsx() subroutine. 671
gpfs_igetfilesetname() subroutine. 673
gpfs_igetstoragepool() subroutine. 675
gpfs_iopen() subroutine 677
gpfs_iopen64() subroutine 679
gpfs_iputattrsx() subroutine 681
gpfs_iread() subroutine 684
gpfs_ireaddir() subroutine 686
gpfs_ireaddir64() subroutine 688
gpfs_ireadlink() subroutine. 690
gpfs_ireadlink64() subroutine 692
gpfs_ireadx() subroutine. 694
gpfs_iscan_t structure 697
gpfs_lib_init() subroutine 698
gpfs_lib_term() subroutine 699
gpfs_next_inode() subroutine 700
gpfs_next_inode64() subroutine 702
gpfs_next_inode_with_xattrs() subroutine 704
gpfs_next_inode_with_xattrs64() subroutine . . . 706
gpfs_next_xattr() subroutine 708
gpfs_opaque_acl_t structure 710
gpfs_open_inodescan() subroutine 711
gpfs_open_inodescan64() subroutine. 714

gpfs_open_inodescan_with_xattrs() subroutine . . 717
gpfs_open_inodescan_with_xattrs64() subroutine 720
gpfs_prealloc() subroutine 723
gpfs_putacl() subroutine. 725
gpfs_quotactl() subroutine 727
gpfs_quotaInfo_t structure 730
gpfs_seek_inode() subroutine 732
gpfs_seek_inode64() subroutine 734
gpfs_stat() subroutine 736
gpfs_stat_inode() subroutine 738
gpfs_stat_inode64() subroutine 740
gpfs_stat_inode_with_xattrs() subroutine 742
gpfs_stat_inode_with_xattrs64() subroutine . . . 744
gpfsFcntlHeader_t structure 746
gpfsGetFilesetName_t structure 747
gpfsGetReplication_t structure. 748
gpfsGetSetXAttr_t structure 750
gpfsGetSnapshotName_t structure 752
gpfsGetStoragePool_t structure 753
gpfsListXAttr_t structure 754
gpfsRestripeData_t structure 755
gpfsSetReplication_t structure 757
gpfsSetStoragePool_t structure. 759

Chapter 4. GPFS user exits 761
mmsdrbackup user exit 762
nsddevices user exit 763
syncfsconfig user exit. 764

Accessibility features for IBM Spectrum Scale 765
Accessibility features 765
Keyboard navigation 765
IBM and accessibility 765

Notices 767
Trademarks 768
Terms and conditions for product documentation 769
IBM Online Privacy Statement. 769

Glossary 771

Index 777

Tables

1. IBM Spectrum Scale library information units ix
2. Conventions xii
3. Documentation guides in this release. . . . xiv
4. Documentation structure in this release xiv
5. GPFS commands 1
6. Global events and supported parameters 15
7. Local events and supported parameters 16
8. Query details by type 33
9. key-value 88
10. key-value 89
11. GPFS commands that support QoS 197
12. mmkeyserv server show 343
13. mmkeyserv tenant show 344
14. Restoring a global snapshot 472
15. Restoring a fileset snapshot 472
16. DMAPI configuration attributes 571
17. Specific DMAPI functions and associated error codes. 601
18. GPFS programming interfaces 607
19. GPFS user exits 761

© Copyright IBM Corp. 2014, 2016

About this information

This edition applies to IBM Spectrum Scale™ version 4.2.1 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
IBM Spectrum Scale: Administration Guide	This information unit explains how to do the following: • 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

© Copyright IBM Corp. 2014, 2016

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

Information unit	Type of information	Intended users
IBM Spectrum Scale: Administration Guide	This information unit explains how to use the following advanced features of GPFS: • 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
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	This information unit provides information about the following topics: • 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
IBM Spectrum Scale: Command and Programming Reference	This information unit describes the Data Management Application Programming Interface (DMAPI) for GPFS. This implementation is based on The Open Group's System Management: Data Storage Management (XDSM) 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. The XDSM 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. Use this information if you intend to write application programs to do the following: • 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 XDSM standard
IBM Spectrum Scale: Problem Determination Guide	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 *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*

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 xii 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	**Bold** words or characters represent system elements that you must use literally, such as commands, flags, values, and selected menu options. Depending on the context, **bold** typeface sometimes represents path names, directories, or file names.
bold underlined	**bold underlined** keywords are defaults. These take effect if you do not specify a different keyword.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface. Depending on the context, `constant-width` typeface sometimes represents path names, directories, or file names.
italic	*Italic* words or characters represent variable values that you must supply. *Italics* are also used for information unit titles, for the first use of a glossary term, and for general emphasis in text.
<*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 *Enter*.

\	In command examples, a backslash indicates that the command or coding example continues on the next line. For example:
	`mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \`
	`-E "PercentTotUsed < 85" -m p "FileSystem space used"`

{*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.
|	In *synopsis* statements, vertical lines separate a list of choices. In other words, a vertical line means *Or*. In the left margin of the document, vertical lines indicate technical changes to the information.

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 2.1
as updated, July 2016

This release of the IBM Spectrum Scale licensed program and the IBM Spectrum Scale library include the following improvements:

Auditing configuration changes
A syslog entry is automatically written whenever a GPFS command makes a configuration change. Adding the information to the syslog gives flexibility in mining, processing, and redirecting these events. Entries can also be written to the standard GPFS log. The **commandAudit** parameter of the **mmchconfig** command controls this option. For more information, see the topic *Audit messages for cluster configuration changes* in the *IBM Spectrum Scale: Problem Determination Guide*.

Automated configuration of sensors for performance monitoring
IBM Spectrum Scale now supports automated configuration of sensors for its performance monitoring tool. For more information on automated configuration of sensors, see the *Automated configuration* section in the IBM Spectrum Scale: Administration Guide.

Callback event for file system structure errors
A new user callback event **fsstruct** (file system structure error) is triggered when the file system detects an error in the metadata. Immediate notification enables the callback program to act to mitigate further errors. For more information, see the topic *mmaddcallback command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

CES, NFS, and SMB protocols: Support for SLES V12 on x86 systems
Cluster Export Services (CES) partially supports SUSE Linux Enterprise Server (SLES) V12 on x86 systems. The SMB and NFS protocols are now supported via a manual installation process. For more information, see the topic *Manually installing IBM Spectrum Scale on SLES 12 systems* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Compression support for FPO environments
File compression is expanded to support the File Placement Optimizer (FPO) environment. For the FPO environment, you must set the block group factor to a multiple of 10 to avoid degrading file system performance. For more information, see the topic *File compression* in the *IBM Spectrum Scale: Administration Guide*.

Deadlock management and debug data control
Deadlock management is extended with the following features:

- The detection thresholds for deadlocks are automatically adjusted according to waiter length and cluster overloaded status.
- New defaults more suitable for customer environments are established for the configuration variables **deadlockDataCollectionDailyLimit**, **deadlockDataCollectionMinInterval**, and others.

A new configuration variable **debugDataControl** controls the amount of debug data that is collected. The default setting is a minimal amount of debug information that is the most important for debugging issues.For more information, see the topic *Managing deadlocks* in the *IBM Spectrum Scale: Problem Determination Guide*.

© Copyright IBM Corp. 2014, 2016

/dev/<fs_name> device for a file system on Linux

On Linux, GPFS no longer creates the /dev/<fs_name> device for a file system. Applications that relied on the file system device under /dev to be present, or that relied on "/dev" to be displayed in the output of the mount command, must find other ways to obtain the information. As a substitute, consider the information provided by the /etc/fstab file and /proc/mounts entries.

Documentation changes

The IBM Spectrum Scale documentation guides are changed as follows:

Table 3. Documentation guides in this release

Guides in release 4.2 and earlier	Guides in release 4.2.1
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	*IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
IBM Spectrum Scale: Administration and Programming Reference Except commands, programming interfaces, and user exits	*IBM Spectrum Scale: Administration Guide*
IBM Spectrum Scale: Advanced Administration Guide	
IBM Spectrum Scale: Problem Determination Guide	*IBM Spectrum Scale: Problem Determination Guide*
IBM Spectrum Scale: Data Management API Guide	*IBM Spectrum Scale: Command and Programming Reference*
IBM Spectrum Scale: Administration and Programming Reference (Commands, programming interfaces, and user exits)	

The top-level structure of the IBM Spectrum Scale documentation in the IBM Knowledge Center is changed as follows:

Table 4. Documentation structure in this release

Documentation structure in release 4.2 and earlier	Documentation structure in release 4.2.1
1. IBM Spectrum Scale: Concepts, Planning, and Installation Guide 2. IBM Spectrum Scale: Administration and Programming Reference 3. IBM Spectrum Scale: Advanced Administration Guide 4. IBM Spectrum Scale: Data Management API Guide 5. IBM Spectrum Scale: Problem Determination Guide	1. Summary of changes 2. Quick reference 3. Product overview 4. Planning 5. Installing and upgrading 6. Configuring 7. Administering 8. Monitoring 9. Troubleshooting 10. Command reference 11. Programming reference 12. Library and related publications 13. Glossary

Encryption: Simplified setup and Vormetric DSM support

- A new console command **mmkeyserv** greatly simplifies the setup of encryption both on the key server and the client node. IBM Security Key Lifecycle Manager (SKLM) V2.5.0.4 or later (including V2.6) is required.
- Encryption support is added for key servers that run Vormetric Data Security Manager (DSM) V5.2.3 or later.

For more information, see the topic *Establishing an encryption-enabled environment* in the *IBM Spectrum Scale: Administration Guide*.

Federation in the performance monitoring tool

A performance monitoring tool installation with multiple collectors is called a federation. Federation is introduced in performance monitoring to increase the scalability or the fault-tolerance of the performance monitoring system. For more information on federation, see the *Configuring multiple collectors* section in the IBM Spectrum Scale: Administration Guide.

Guided installation

The **spectrumscale** installation toolkit now provides next step hints that are designed to help customers new to IBM Spectrum Scale with an easy workflow that helps customers to install and configure an IBM Spectrum Scale cluster.

Hadoop Support for IBM Spectrum Scale

HDFS transparency now supports running the Hadoop Map/Reduce workload inside the virtual machine container, Docker.

Federation is introduced in HDFS to solve the HDFS NameNode scaling problem.

Hadoop distcp is used for data migration from HDFS to the IBM Spectrum Scale file system and between two IBM Spectrum Scale file systems.

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

- *Docker support*
- *The HDFS transparency federation*
- *Hadoop distcp support*

HDFS transparency security has been introduced for the simple security mode and the Kerberos mode.
User authentication and authorization is weak in the simple mode. The data transfers and RPCs from the clients to the NameNode and DataNode are not encrypted. The Kerberos mode introduced in the Hadoop ecosystem provides a secure Hadoop environment.
For more information, see the *HDFS transparency security* section in the *IBM Spectrum Scale: Administration Guide*.

InfiniBand and RDMA performance

Performance is improved for clusters that use InfiniBand and RDMA for their intranode communications network.

Linux on z Systems™: Expanded features

The following IBM Spectrum Scale features are now available on Linux for z Systems:

- Quality of Service (QoS) support.
- Improved extended count key data (ECKD™) device handling: On different nodes, different bus IDs can refer to the same device.
- IBM Spectrum Scale GUI now supported on Linux for z Systems.

--metadata-only parameter for mmrestripefs

A **--metadata-only** option for the **mmrestripefs** command allows the restripe to complete in less time than a full restripe of metadata and data. The savings in time is useful in situations where there is a concern about file system operations and you want to restripe. This operation is supported for migrating data off disks, rebalancing, restoring replication, and comparing replicas.
For more information, see the topic *mmrestripefs command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

mmhealth: Monitoring services hosted on cluster nodes

A new command, **mmhealth** is added to monitor the health status of nodes and different services hosted on nodes. The **mmhealth** command also displays the event logs responsible for the

unhealthy status of nodes and services, to analyze and determine the problem responsible for the service failure. For more information, see *mmhealth command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Object storage improvements

- Added support for starting and stopping the ibmobjectizer service. For information about ibmobjectizer service, see *Starting and Stopping the ibmobjectizer service* in *IBM Spectrum Scale: Administration Guide*.
- For problem determination, added potential problem scenarios with proposed solutions. The problem determination scenarios are listed here: *Object issues* in *IBM Spectrum Scale: Problem Determination Guide*.
- Added support for object encryption. For information about object encryption, see *Creating storage policy for encryption* in *IBM Spectrum Scale: Administration Guide*.
- Added new constraints for unified file and object access. The constraints are listed here: *Constraints applicable to unified file and object access* in *IBM Spectrum Scale: Administration Guide*.
- Added support for simplified enablement of S3. For information on S3, see *Changing the object base configuration to enable S3 API* in *IBM Spectrum Scale: Administration Guide*.
- Added support for multi-region object deployment with a highly available keystone service. For information about the multi-region object deployment, see *Authentication considerations for multi-region object deployment* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Added support for OpenStack Liberty packages. For more information on Liberty packages, see *Protocol support overview: Integration of protocol access methods with GPFS* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Added support to execute mmobj commands from any IBM Spectrum Scale client node. For more information on mmobj, see *mmobj command* in *IBM Spectrum Scale: Command and Programming Reference*.
- Added support for monitoring support for external AD and LDAP server for object authentication and main object services. For more information on external AD and LDAP server, see *Configuring an AD-based authentication for object access* in *IBM Spectrum Scale: Administration Guide*.

Quality of Service for I/O operations (QoS) improvements

Quality of Service for I/O operations is expanded to support the File Placement Optimizer (FPO) environment. For more information, see the topic *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Administration Guide*.

Re-create and restore options for protocols cluster failover

The failover procedure can choose between re-create and restore options. For more information on the failover options, see the *Performing failover for protocols cluster when primary cluster fails* section in the IBM Spectrum Scale: Administration Guide.

Re-create and restore options for failing back to an old primary for protocols cluster

When failing back to an old primary, the file protocol configuration can either be re-created or restored. For more information on failing back to an old primary, see the *Performing failback to old primary for protocols cluster* section in the IBM Spectrum Scale: Administration Guide.

Re-create and restore options for failing back to a new primary for protocols cluster

When failing back to a new primary, the file protocol configuration can either be re-created or restored. For more information on failing back to an old primary, see the *Performing failback to new primary for protocols cluster* section in the IBM Spectrum Scale: Administration Guide.

Support for Transparent Cloud Tiering

The Transparent Cloud Tiering feature leverages the existing ILM policy available in IBM Spectrum Scale, and administrators can define policies to migrate cold data to a cloud storage tier or recall data from the cloud storage tier on reaching certain threshold levels.

A new command, **mmcloudgateway**, is added to manage and configure the cloud storage tier.

Note: To enable Transparent Cloud Tiering nodes, you must first enable the Transparent Cloud Tiering feature. This feature provides a new level of storage tiering capability to IBM Spectrum Scale customers. Please contact your IBM Client Technical Specialist (or send an email to mailto:scale@us.ibm.com) to review your use case of the Transparent Cloud Tiering feature and to obtain the instructions to enable the feature in your environment.

workerThreads tunes file system performance

The **workerThreads** parameter of the **mmchconfig** command 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. This variable can be used in any installation and is preferred over **worker1Threads** and **prefetchThreads** in new installations. For more information, see the topic *mmchconfig command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

IBM Spectrum Scale GUI changes

The following main changes are added in the IBM Spectrum Scale management GUI:

- Renamed Monitoring > Topology page to NSDs. The NSDs page facilitates monitoring the status of Network Shared Disks (NSD) and nodes to NSD mapping in the system.
- Added new Monitoring > Nodes page in the GUI. The Nodes page provides an easy way to monitor the performance, health status, and configuration aspects of all available nodes in the IBM Spectrum Scale cluster. The properties of a node display the status of various CES servicesrecr such as Object, NFS, and SMB as well as the authentication status of these services if they are enabled. It also displays other details such as network status, information on attached NSDs and file systems, and so on.
- Monitoring performance of transparent cloud tiering services through Performance and Dashboard pages.
- Renamed Monitoring > Performance page to Statistics.
- Added capacity monitoring options in the Statistics page.
- Added monitoring options for GPFS waiters in Monitoring > Statistics panel.
- The following improvements are made in the Dashboards page:
  - You can assign a name to the dashboards and the user can switch between dashboards.
  - The dashboards are now stored on the server instead of the browser. Therefore, it can be shared among users and browsers.
  - Default dashboards are shipped with the GUI. When you open the IBM Spectrum Scale™ GUI after the installation or upgrade, you can see the default dashboards. You can further modify or delete the default dashboards to suit your requirements.
- Renamed Download Logs page as Diagnostic Data. Now, the GUI can be used instead of the **gpfs.snap** command to collect the details of the issue. For more information on collecting diagnostic data through GUI, see *Collecting diagnostic data through GUI* topic in the *IBM Spectrum Scale: Problem Determination Guide*.
- The Files > Information Lifecycle page facilitates defining compression and deletion rules.
- The new Settings > Object Service page facilitates start and stop feature for object services.
- Up to 1000 nodes are supported.
- The GUI can now be used an IBM Spectrum Scale cluster where sudo wrappers are used. For more information on how to configure IBM Spectrum Scale GUI to use sudo wrapper, see *Configuring IBM Spectrum Scale GUI to use sudo wrapper* in *IBM Spectrum Scale: Administration Guide*.
- IBM Spectrum Scale GUI support for System z® platform is available on RHEL7.2 and SLES12.
- By default, GUI commands that change the configuration of the cluster cause an audit message to be sent to syslog. Optionally, an audit message can also be sent to the GPFS log. For more information, see the topic *Audit messages for cluster configuration changes* in the *IBM Spectrum Scale: Problem Determination Guide*.

NFS and SMB protocol troubleshooting information added

New AD Discovery tool to query and validate several AD settings.

New troubleshooting information for NFS issues:

- NFS mount issues
- NFS error events
- NFS error scenarios

New troubleshooting information for SMB issues:

- SMB client on Linux failures
- SMB mount errors
- SMB error events
- SMB access issues

Documented commands, structures, and subroutines

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

New commands

The following commands are new:

- **mmadquery**
- **mmcloudgateway**
- **mmhealth**
- **mmkeyserv**

New structures

There are no new structures.

New subroutines

There are no new subroutines.

Changed commands

The following commands were changed:

- **gpfs.snap**
- **mmafmlocal**
- **mmcallhome**
- **mmchconfig**
- **mmchnode**
- **mmcesdr**:
- **mmcrsnapshot**: You can create multiple snapshots in the same command.
- **mmdelsnapshot**: You can delete multiple snapshots in the same command.
- **mmlscluster**
- **mmnfs**
- **mmobj**
- **mmrestripefs**
- **mmsmb**
- **mmprotocoltrace**
- **mmsmb**
- **mmuserauth**

Changed structures

There are no changed structures.

Changed subroutines

gpfs_iopen() subroutine

gpfs_iopen64() subroutine

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

6027-1826, 6027-2363, 6027-2364, 6027-2365, 6027-2366, 6027-2367, 6027-2368, 6027-2369, 6027-2370, 6027-2371, 6027-2372, 6027-2373, 6027-2374, 6027-2375, 6027-2376, 6027-2377, 6027-2378, 6027-3108, 6027-3720, 6027-3721, 6027-3722, 6027-3723, 6027-3724, 6027-3725, 6027-3726, 6027-3727, 6027-3728, 6027-3915, 6027-3916, 6027-3594, 6027-3595, 6027-3596

Changed messages

6027-1368, 6027-1235, 6027-1545, 6027-2271, 6027-2272, 6027-2273, 6027-2274, 6027-2951

Deleted messages

6027-1997

Chapter 1. Command reference

A list of all the GPFS commands and a short description of each is presented in this topic.

Table 5 summarizes the GPFS-specific commands.

Table 5. GPFS commands

Command	Purpose
"gpfs.snap command" on page 5	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 10	Registers a user-defined command that GPFS will execute when certain events occur.
"mmadddisk command" on page 23	Adds disks to a GPFS file system.
"mmaddnode command" on page 29	Adds nodes to a GPFS cluster.
"mmadquery command" on page 32	
"mmafmconfig command" on page 37	Can be used to manage home caching behavior and mapping of gateways and home NFS exported servers.
"mmafmctl command" on page 40	This command is for various operations and reporting information on all filesets. It is recommended to read the *IBM Spectrum Scale: Administration Guide* AFM and AFM Disaster Recovery chapters in conjunction with this manual for detailed description of the functions.
"mmafmlocal command" on page 54	Provides a list of cached files and file statistics such as inode number, allocated blocks, and so on.
"mmapplypolicy command" on page 56	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 67	Manages secure access to GPFS file systems.
"mmbackup command" on page 72	Performs a backup of a GPFS file system or independent fileset to an IBM Spectrum Protect™ server.
"mmbackupconfig command" on page 81	Collects GPFS file system configuration information.
"mmbuildgpl command" on page 83	Manages prerequisite packages for Linux and builds the GPFS portability layer.
"mmcallhome command" on page 85	Manages the call home operations.
"mmces command" on page 96	Manages CES configuration.
"mmcesdr command" on page 105	Manages protocol cluster disaster recovery.
"mmchattr command" on page 113	Changes attributes of one or more GPFS files.
"mmchcluster command" on page 119	Changes GPFS cluster configuration data.
"mmchconfig command" on page 123	Changes GPFS configuration parameters.
"mmchdisk command" on page 151	Changes state or parameters of one or more disks in a GPFS file system.
"mmcheckquota command" on page 159	Checks file system user, group and fileset quotas.
"mmchfileset command" on page 163	Changes the attributes of a GPFS fileset.
"mmchfs command" on page 169	Changes the attributes of a GPFS file system.

© Copyright IBM Corporation © IBM 2014, 2016

Table 5. GPFS commands (continued)

Command	Purpose
"mmchlicense command" on page 175	Controls the type of GPFS license associated with the nodes in the cluster.
"mmchmgr command" on page 178	Assigns a new file system manager node or cluster manager node.
"mmchnode command" on page 180	Changes node attributes.
"mmchnodeclass command" on page 185	Changes user-defined node classes.
"mmchnsd command" on page 188	Changes Network Shared Disk (NSD) configuration attributes.
"mmchpolicy command" on page 191	Establishes policy rules for a GPFS file system.
"mmchpool command" on page 194	Modifies storage pool properties.
"mmchqos command" on page 196	Changes the Quality of Service for I/O operations (QoS) settings for a file system.
"mmclone command" on page 201	Creates and manages file clones.
"mmcloudgateway command" on page 204	Creates and manages the cloud storage tier.
"mmcrcluster command" on page 218	Creates a GPFS cluster from a set of nodes.
"mmcrfileset command" on page 223	Creates a GPFS fileset.
"mmcrfs command" on page 229	Creates a GPFS file system.
"mmcrnodeclass command" on page 239	Creates user-defined node classes.
"mmcrnsd command" on page 241	Creates Network Shared Disks (NSDs) used by GPFS.
"mmcrsnapshot command" on page 246	Creates a snapshot of a file system or fileset at a single point in time.
"mmdefedquota command" on page 251	Sets default quota limits.
"mmdefquotaoff command" on page 254	Deactivates default quota limit usage.
"mmdefquotaon command" on page 257	Activates default quota limit usage.
"mmdefragfs command" on page 260	Reduces disk fragmentation by increasing the number of full free blocks available to the file system.
"mmdelacl command" on page 263	Deletes a GPFS access control list.
"mmdelcallback command" on page 265	Deletes one or more user-defined callbacks from the GPFS system.
"mmdeldisk command" on page 266	Deletes disks from a GPFS file system.
"mmdelfileset command" on page 271	Deletes a GPFS fileset.
"mmdelfs command" on page 274	Removes a GPFS file system.
"mmdelnode command" on page 276	Removes one or more nodes from a GPFS cluster.
"mmdelnodeclass command" on page 279	Deletes user-defined node classes.
"mmdelnsd command" on page 281	Deletes Network Shared Disks (NSDs) from the GPFS cluster.
"mmdelsnapshot command" on page 283	Deletes a GPFS snapshot.
"mmdf command" on page 287	Queries available file space on a GPFS file system.
"mmdiag command" on page 290	Displays diagnostic information about the internal GPFS state on the current node.
"mmeditacl command" on page 295	Creates or changes a GPFS access control list.
"mmedquota command" on page 298	Sets quota limits.
"mmexportfs command" on page 302	Retrieves the information needed to move a file system to a different cluster.

Table 5. GPFS commands (continued)

Command	Purpose
"mmfsck command" on page 304	Checks and repairs a GPFS file system.
"mmfsctl command" on page 313	Issues a file system control request.
"mmgetacl command" on page 317	Displays the GPFS access control list of a file or directory.
"mmgetstate command" on page 320	Displays the state of the GPFS daemon on one or more nodes.
"mmhadoopctl command" on page 323	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 329	Performs a backup of a single GPFS file system metadata image.
"mmimgrestore command" on page 333	Restores a single GPFS file system from a metadata image.
"mmimportfs command" on page 336	Imports into the cluster one or more file systems that were created in another GPFS cluster.
"mmkeyserv command" on page 340	Manages encryption key servers and clients.
"mmlinkfileset command" on page 350	Creates a junction that references the root directory of a GPFS fileset.
"mmlsattr command" on page 352	Queries file attributes.
"mmlscallback command" on page 355	Lists callbacks that are currently registered in the GPFS system.
"mmlscluster command" on page 357	Displays the current configuration information for a GPFS cluster.
"mmlsconfig command" on page 359	Displays the current configuration data for a GPFS cluster.
"mmlsdisk command" on page 361	Displays the current configuration and state of the disks in a file system.
"mmlsfileset command" on page 365	Displays attributes and status for GPFS filesets.
"mmlsfs command" on page 369	Displays file system attributes.
"mmlslicense command" on page 373	Displays information about the GPFS node licensing designation.
"mmlsmgr command" on page 375	Displays which node is the file system manager for the specified file systems or which node is the cluster manager.
"mmlsmount command" on page 377	Lists the nodes that have a given GPFS file system mounted.
"mmlsnodeclass command" on page 379	Displays node classes defined in the system.
"mmlsnsd command" on page 381	Displays Network Shared Disk (NSD) information for the GPFS cluster.
"mmlspolicy command" on page 384	Displays policy information.
"mmlspool command" on page 386	Displays information about the known storage pools.
"mmlsquota command" on page 391	Displays quota information for a user, group, or fileset.
"mmlsqos command" on page 388	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 395	Displays GPFS snapshot information.
"mmmigratefs command" on page 398	Performs needed conversions to support new file system features.
"mmmount command" on page 400	Mounts GPFS file systems on one or more nodes in the cluster.

Table 5. GPFS commands (continued)

Command	Purpose
"mmnfs command" on page 402	Manages NFS exports and configuration.
"mmnsddiscover command" on page 412	Rediscovers paths to the specified network shared disks.
"mmobj command" on page 414	Manages configuration of Object protocol service, and administers storage policies for object storage, unified file and object access, and multi-region object deployment.
"mmperfmon command" on page 428	Configures the Performance Monitoring tool and lists the performance metrics.
"mmpmon command" on page 438	Manages performance monitoring and displays performance information.
"mmprotocoltrace command" on page 443	Starts, stops, and monitors tracing for the CES protocols.
"mmpsnap command" on page 447	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 450	Sets the GPFS access control list for the specified file or directory.
"mmquotaoff command" on page 453	Deactivates quota limit checking.
"mmquotaon command" on page 455	Activates quota limit checking.
"mmremotecluster command" on page 457	Manages information about remote GPFS clusters.
"mmremotefs command" on page 460	Manages information needed for mounting remote GPFS file systems.
"mmrepquota command" on page 463	Displays file system user, group, and fileset quotas.
"mmrestoreconfig command" on page 467	Restores file system configuration information.
"mmrestorefs command" on page 471	Restores a file system or an independent fileset from a snapshot.
"mmrestripefile command" on page 475	Rebalances or restores the replication factor of the specified files, or performs any incomplete or deferred file compression or decompression.
"mmrestripefs command" on page 478	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 485	Replaces the specified disk.
"mmsdrrestore command" on page 492	Restores the latest GPFS system files on the specified nodes.
"mmsetquota command" on page 494	Sets quota limits.
"mmshutdown command" on page 498	Unmounts all GPFS file systems and stops GPFS on one or more nodes.
"mmsmb command" on page 500	Administers SMB shares, export ACLs, and global configuration.
"mmsnapdir command" on page 511	Controls how the special directories that connect to snapshots appear.
"mmstartup command" on page 515	Starts the GPFS subsystem on one or more nodes.
"mmtracectl command" on page 517	Sets up and enables GPFS tracing.
"mmumount command" on page 521	Unmounts GPFS file systems on one or more nodes in the cluster.
"mmunlinkfileset command" on page 524	Removes the junction to a GPFS fileset.

Table 5. GPFS commands (continued)

Command	Purpose
"mmuserauth command" on page 527	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 546	Manages the **mmwinserv** Windows service.
"spectrumscale command" on page 548	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 IBM Spectrum Scale 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]
```

```
[--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.

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 *Specifying nodes as input to GPFS commands* in *IBM Spectrum Scale: Administration Guide*.

--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, when the Transparent Cloud Tiering service is enabled, the snap will collect information such as logs, traces, Java™ cores, along with minimal system and IBM Spectrum Scale cluster information specific to transparent cloud tiering. No customer sensitive information is collected.

Note: The default behavior of the **gpfs.snap** command includes basic information of Transparent Cloud Tiering, in addition to the GPFS information.

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

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 Guide*.

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

2. 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 IBM Spectrum Scale 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 6 on page 15 lists the supported global events and their parameters.

  Table 7 on page 16 lists the supported local events and their parameters.

  Local events for IBM Spectrum Scale 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 *IBM Spectrum Scale: Administration Guide*.

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.

%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.

%filesetName
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.

%nodeNames
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.

%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 IBM Spectrum Scale 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 6. 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**

Table 6. 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 7. Local events and supported parameters

Local event	Supported parameters
afmCmdRequeued Triggered during replication when messages are queued up again because of errors. These messages are retried after 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 completed. This event is local on a gateway node.	**%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 7. 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 **/usr/lpp/mmfs/samples/deadlockdetected.sample** 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**

Table 7. Local events and supported parameters (continued)

Local event	Supported parameters
fsstruct Triggered when the file system manager detects a file system structure (FS Struct) error. For more information about FS Struct errors, see "mmfsck command" on page 304 and the following topics in *IBM Spectrum Scale: Problem Determination Guide*: • *MMFS_FSSTRUCT* • *Reliability, Availability, and Serviceability (RAS) events* • *Information to collect before contacting the IBM Support Center*	**%fsName %fsErr %senseDataFormatted %senseDataHex**
healthCollapse Triggered when the node health declines below the **healthCollapseThreshold** long enough for the health check thread to notice.	N/A
lowDiskSpace 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 *Using thresholds with external pools* in the *IBM Spectrum Scale: Administration Guide*.	**%storagePool %fsName**
noDiskSpace 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. 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". 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".	**%storagePool %fsName %reason %filesetName**

Table 7. Local events and supported parameters (continued)

Local event	Supported parameters
nsdCksumMismatch 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.	**%myNode %ckRole %ckOtherNode %ckNSD %ckReason %ckStartSector %ckDataLen %ckErrorCountClient %ckErrorCountServer %ckErrorCountNSD %ckReportingInterval**
pdFailed The **pdFailed** callback is generated whenever a pdisk in a recovery group is marked as `dead`, `missing`, `failed`, or `readonly`.	**%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdState**
pdPathDown 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.	**%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdPath**
pdReplacePdisk 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.	**%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn %pdState %pdPriority**
pdRecovered The **pdRecovered** callback is generated whenever a missing pdisk is rediscovered. The following parameters are available to this callback: **%myNode**, **%rgName**, **%daName**, **%pdName**, **%pdLocation**, **%pdFru**, and **%pdWwn**.	**%myNode %rgName %daName %pdName %pdLocation %pdFru %pdWwn**
preMount, preUnmount, mount, unmount 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.	**%fsName %reason**
preRGRelinquish 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**.	**%myNode %rgName %rgErr %rgCount %rgReason**

Table 7. Local events and supported parameters (continued)

Local event	Supported parameters
preRGTakeover 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**.	**%myNode %rgName %rgErr %rgCount %rgReason**
preShutdown Triggered when GPFS detects a failure and is about to shut down.	**%reason**
preStartup 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.	N/A
postRGRelinquish 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.	**%myNode %rgName %rgErr %rgCount %rgReason**
postRGTakeover 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.	**%myNode %rgName %rgErr %rgCount %rgReason**
rgOpenFailed 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.	**%myNode %rgName %rgErr %rgReason**

Table 7. Local events and supported parameters (continued)

Local event	Supported parameters
rgPanic 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.	**%myNode %rgName %rgErr %rgReason**
sendRequestToNodes Triggered when a node sends a request for collecting expel-related debug data. For this event, the **%requestType** is **requestExpelData**.	**%eventName %requestType %nodeNames**
shutdown Triggered when GPFS completes the shutdown.	**%reason**
snapshotCreated 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. 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**.	**%snapshotID %snapshotName %fsName %filesetName**
softQuotaExceeded 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.	**%fsName %filesetName %quotaId %quotaType %quotaOwnerName %blockUsage %blockQuota %blockLimit %filesUsage %filesQuota %filesLimit**
startup 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.	N/A
tiebreakerCheck 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.	N/A
traceConfigChanged Triggered when GPFS tracing configuration is changed.	N/A

Table 7. 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 nfsserver1,
nfsserver2 --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 265
- "mmlscallback command" on page 355

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:

   ```
   mmlsnsd -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.

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 **mmlsnsd -F** command. This clause is mandatory for the **mmadddisk** 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: 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 **mmadddisk** command.

device=*DiskName*
The block device name of the underlying disk device. This clause is ignored by the **mmadddisk** 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

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 File Placement Optimizer (FPO) feature is to be enabled for the storage pool. For more information on FPO, see the *File Placement Optimizer* section in the *IBM Spectrum Scale: Administration Guide*.

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. The following conditions must be met while using a write affinity depth of 2 to get evenly allocated space in all disks:

1. The configuration in disk number, disk size, and node number for each rack must be similar.
2. The number of nodes must be the same in the bottom half and the top half of each rack.

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 *File Placement Optimizer* in *IBM Spectrum Scale: Administration Guide*.

-a Specifies asynchronous processing. If this flag is specified, the **mmadddisk** 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 Guide*.

--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" on page 196. 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: Administration Guide*.

Exit status

0 Successful completion.

nonzero
A failure occurred.

Security

You must have root authority to run the **mmadddisk** 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmadddisk 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
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.
mmadddisk: 6027-1371 Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
Restriping 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 151
- "mmcrnsd command" on page 241
- "mmdeldisk command" on page 266
- "mmlsdisk command" on page 361
- "mmlsnsd command" on page 381
- "mmlspool command" on page 386

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.

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 Guide*.

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:

```
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:    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 123
- "mmcrcluster command" on page 218
- "mmchcluster command" on page 119
- "mmdelnode command" on page 276
- "mmlscluster command" on page 357

Location

```
/usr/lpp/mmfs/bin
```

mmadquery command

Queries and validates Active Directory (AD) server settings.

Synopsis

mmadquery list {**user** | **uids** | **gids** | **groups** | **dc** | **trusts** | **idrange**} [*Options*]

or

mmadquery check {**uids** | **gids** | **idrange**} [*Options*]

or

mmadquery stats {**user** |**uids**}

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmadquery** command to query an AD Server for users, groups, user IDs, group IDs, known domain controller and trusts, and to run consistency checks.

Parameters

user
Queries and lists the defined users.

uids
Queries and lists the defined users with user IDs and group IDs.

gids
Queries and lists the defined groups with group IDs.

groups
Queries and lists the defined groups.

dc Queries and lists the defined domain controllers.

trusts
Queries and lists the defined trusts.

idrange
Queries and lists the ID range used by a given AD server.

Options

--server *SERVER*
Specifies the IP address of the AD server you want to query. If you do not specify a server, **mmadquery** attempts to get the AD server from the/etc/resolv.conf file (name server).

--domain *DOMAIN*
Specifies the Windows domain. If you do not specify a domain, **mmadquery** uses **nslookup** to determine the domain based on the server.

--user *USER*
Specifies the user for the AD server query. The default is Administrator.

--pwd-file *File*
Specifies the file that contains a password to use for authentication.

--filter *FILTER*
Specifies any string to filter the query output.

--CSV
Shows output in machine parsable (CSV) format.

--debug or -d
Shows debugging information

--basedn or -b
Includes basedn in query output.

--traverse
Traverses all known domains and provide query output for all domains that are detected.

--long or L
Indicates that you want to see more details. For more information, see Level of detail below.

Level of query detail

Table 8. Query details by type

Query	Additional content
User	Group membership
DC	Operating system
UIDs	GID, Primary Group ID
Trusts	DC

Exit status

0 No errors found.

1 No arguments specified.

10 Failed a check.

11 Unable to determine the AD server to check.

12 Unable to determine the domain.

13 Failed to construct a basedn for an LDAP query.

99 Access to the AD server failed, can be incorrect password, user, or domain.

Security

You must have root authority to run the mmadquery command. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

Examples

1. To show a list of users for the AD server, run this command:

```
mmadquery list user --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
USER from server 9.155.106.234 (domain subdom1.mzdom.com)
          User
 -------------
  Administrator
          Guest
         krbtgt
         MZDOM$
        aduser1
        aduser2
        Taduser3
```

2. To show a list of groups for the AD server, run this command:

```
mmadquey list groups --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
GROUPS from server 9.155.106.234 (domain subdom1.mzdom.com)
                                        Group
      ---------------------------------------
                               Domain Computers
                                Cert Publishers
                                   Domain Users
                                  Domain Guests
                             RAS and IAS Servers
                                   Domain Admins
                                   Schema Admins
                                Enterprise Admins
                     Group Policy Creator Owners
           Allowed RODC Password Replication Group
            Denied RODC Password Replication Group
           Enterprise Read-only Domain Controllers
                               Domain Controllers
                  Read-only Domain Controllers
                                         DnsAdmins
                                    DnsUpdateProxy
                                           UNIXGRP
                                    unmapped group
                                               bla
```

3. To check user IDs against locally defined ID mapping range, issue the following command:

```
mmadquery check uids --pwd-file /tmp/mmadquery.cfg -L
```

The system displays information similar to:

```
UIDS from server 9.155.106.234 (domain subdom1.mzdom.com)
       User   SID                                        UID  UIDNumber GIDNumber Primary Group ID
 ------------------------------------------------------------------------------------------------
Guest          S-1-5-21-2808815044-4164012579-2832416960-501  -   -         -           514
SUBDOM1$       S-1-5-21-2808815044-4164012579-2832416960-1103 -   -         -           513
Administrator  S-1-5-21-2808815044-4164012579-2832416960-500  -   -         -           513
       krbtgt  S-1-5-21-2808815044-4164012579-2832416960-502  -   -         -           513
       User 1  S-1-5-21-2808815044-4164012579-2832416960-1107 -   -         -           513
      aduser1  S-1-5-21-2808815044-4164012579-2832416960-1601 aduser1 20000007 20000008 513
       User 2  S-1-5-21-2808815044-4164012579-2832416960-1110 aduser 10001    20000009 513
WARNING: UID of user User 2 outside id mapping range 'mzdom'.
```

4. To show a list of users with group membership by domain, run this command:

```
mmadquery list user -L --pwd-file /tmp/mmadquery.cfg --traverse
```

The system displays information similar to:

```
USER from server 9.155.106.232 (domain mzdom.com)
       User                                                                                  Groups
       ------------------------------------------------------------------------------------  ---------
       Guest                                                                                 Guests

       SUBDOM1$
Administrator Group Policy Creator Owners,Enterprise Admins,Schema Admins,Domain Admins,Administrators
       krbtgt                                                 Denied RODC Password Replication Group
       aduser1                                                                    Administrators
       aduser2                                                                 bla,unmapped group
       aduser3
       aduser4

USER from server 9.155.106.234 (domain subdom1.mzdom.com)
       User                                          Groups
       ------------ ------------------------------------------------------------
       Administrator Group Policy Creator Owners,Domain Admins,Administrators
       Guest                                                 Guests
       krbtgt           Denied RODC Password Replication Group Administrators
       MZDOM$
       aduser1
       aduser2
       aduser3
       aduser4
```

5. To show the number of users by group and domain, run this command:

```
mmadquery stats user -L --pwd-file /tmp/mmadquery.cfg --traverse
```

The system displays information similar to:

```
USER from server 9.155.106.232 (domain mzdom.com)
                                          Group Count
          ------------------------------------- -----
                                           TOTAL     7
                                          Guests     1
                     Group Policy Creator Owners     1
                               Enterprise Admins     1
                                   Schema Admins     1
                                   Domain Admins     1
                                  Administrators     2
             Denied RODC Password Replication Group  1
                                             bla     1
                                  unmapped group     1
USER from server 198.51.100.13 (domain subdom1.mzdom.com)
                                          Group Count
          ------------------------------------- -----
                                           TOTAL     7
                     Group Policy Creator Owners     1
                                   Domain Admins     1
                                  Administrators     2
                                          Guests     1
          Denied RODC Password Replication Group     1
```

6. To show a list of the number of unmapped users, run this command:

```
mmadquery stats uids --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
UIDS from server 9.155.106.232 (domain mzdom.com)
Group      Count
---------  -----
TOTAL      7
MAPPED     2
UN-MAPPED  5
```

7. To check group IDs against locally defined ID map, run this command:

```
mmadquery check gids -L --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

GIDS from server 9.155.106.232 (domain w2k8r2-dom02.mzdom.com)

```
GIDS from server 9.155.106.232 (domain w2k8r2-dom02.mzdom.com)
                                Group                                                SID UID  UIDNumber         GIDNumber
 ---------------------------------------------------------------------------------------------------------------------------
                      Domain Computers S-1-5-21-2808815044-4164012579-2832416960-515      -          -                  -
                       Cert Publishers S-1-5-21-2808815044-4164012579-2832416960-517      -          -                  -
                          Domain Users S-1-5-21-2808815044-4164012579-2832416960-513      -          -  20000008        -
                         Domain Guests S-1-5-21-2808815044-4164012579-2832416960-514      -          -                  -
                   RAS and IAS Servers S-1-5-21-2808815044-4164012579-2832416960-553      -          -                  -
                   Domain Admins       S-1-5-21-2808815044-4164012579-2832416960-512      -          -                  -
                   Schema Admins       S-1-5-21-2808815044-4164012579-2832416960-518      -          -                  -
                   Enterprise Admins   S-1-5-21-2808815044-4164012579-2832416960-519      -          -                  -
            Group Policy Creator Owners S-1-5-21-2808815044-4164012579-2832416960-520      -          -                  -
Allowed RODC Password Replication Group S-1-5-21-2808815044-4164012579-2832416960-571      -          -                  -
Denied RODC Password Replication Group  S-1-5-21-2808815044-4164012579-2832416960-57       -          -                  -
Enterprise Read-only Domain Controllers S-1-5-21-2808815044-4164012579-2832416960-498      -          -                  -
                     Domain Controllers S-1-5-21-2808815044-4164012579-2832416960-516      -          -                  -
Read-only Domain Controllers   S-1-5-21-2808815044-4164012579-2832416960-521              -          -                  -
                   DnsAdmins   S-1-5-21-2808815044-4164012579-2832416960-1101             -          -                  -
              DnsUpdateProxy   S-1-5-21-2808815044-4164012579-2832416960-1102             -          -                  -
                     UNIXGRP   S-1-5-21-2808815044-4164012579-2832416960-1104             -          - 200002222        -
              unmapped group   S-1-5-21-2808815044-4164012579-2832416960-1603             -          -                  -
                         bla   S-1-5-21-2808815044-4164012579-2832416960-1604             -          -                  -
-WARNING: GID of group 'UNIXGRP' outside id mapping range 'mzdom'.
```

8. To show a list of domain controllers, run the following command:

```
mmadquery list dc L --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
DC from server 9.155.106.232 (domain w2k8r2-dom02.mzdom.com)
      DC              Hostname                Operating System
------------- --------------------  ------------------------------
WW2K8R2-DOM03 w2k8r2-dom03.mzdom.com Windows Server 2008 R2 Standard
WW2K8R2-DOM02 w2k8r2-dom02.mzdom.com Windows Server 2008 R2 Standard
```

9. To show a list of trusts, run the following command:

```
mmadquery list trusts --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
TRUSTS from server 9.155.106.232 (domain w2k8r2-dom02.mzdom.com)
       DC                       Trust Type
-----------------     ---------------------------
subdom1.mzdom.com     Within Forest bi-directional
     w2k12dom.com       Forest Transitive outbound
```

10. To show a list of ID ranges and to check whether any IDs on the Ad server are outside of the locally defined ID range, run this command:

```
mmadquery check idrange --pwd-file /tmp/mmadquery.cfg
```

The system displays information similar to:

```
IDRANGE from server 9.155.106.232 (domain w2k8r2-dom02.mzdom.com)
 Domain       IDRange        IDMapRange
--------- -------------- -----------------
msdom.com 10001-200000000 20000000-25999999
WARNING: IDs from domain 'mzdom.com' are outside locally defined id mapping range 'mzdom'.
```

11. To show a list of ID ranges by domain, run this command:

```
mmadquery list idrange --pwd-file /tmp/mmadquery.cfg -L --traverse
```

The system displays information similar to:

```
IDRANGE from server 9.155.106.232 (domain mzdom.com)
   Domain               IDRange        IDMapRange
---------------- ----------------- -----------------
mzdom.com         10001--260000009   10000000-29999999

IDRANGE from server 9.155.106.234 (domain subdom1.mzdom.com)

   Domain               IDRange        IDMapRange
---------------- ----------------- -----------------
subdom1.mzdom.com 200000001-26000010 10000000-29999999
```

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 data transfers 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 data transfers.

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 data transfers. Run at cache only.

delete
Deletes maps for parallel data transfers. Run at cache only.

update
Updates maps for parallel data transfers. 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. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

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:
   ```
   # mmafmconfig add mapping1 --export-map js22n01/hs22n18,js22n02/hs22n19
   ```
   The system displays output similar to: `mmafmconfig: Command successfully completed.`
2. Issue the following command:
   ```
   # mmafmconfig add mapping2 --export-map js22n02/hs22n20,js22n01/hs22n21
   ```
   The system displays output similar to: `mmafmconfig: Command successfully completed.`
   mmafmconfig: Propagating the cluster configuration data to all affected nodes. This is an asynchronous process.
3. Issue the following command:
   ```
   # mmafmconfig 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`

```
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 40
- "mmafmlocal command" on page 54
- "mmchconfig command" on page 123
- "mmchfileset command" on page 163
- "mmchfs command" on page 169
- "mmcrfileset command" on page 223
- "mmcrfs command" on page 229
- "mmlsconfig command" on page 359
- "mmlsfileset command" on page 365
- "mmlsfs command" on page 369

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: Administration Guide* AFM and AFM Disaster Recovery chapters in conjunction with this manual for detailed description of the functions.

Synopsis

To use the AFM DR functions correctly, use all commands enlisted in this chapter in accordance with the steps described in the AFM-based DR chapter in *IBM Spectrum Scale: Concepts, Planning, and Installation 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 | resumeRequeued} [-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 ...]
           [--list-file ListFile] [--file FilePath]
```

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 *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
Primary gateway of the fileset. This gateway node is handling requests for this fileset.

Queue Length
Current length of the queue on the primary gateway.

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`

For more information on all cache states see the AFM and AFM-based DR chapters in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

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. *ListFile* contains a list of files that you want to flush, one file per line. All files must have absolute path names, specified from the fileset linked path. 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. FlushPending is applicable for SW/IW and primary filesets.

4. This section describes:

```
mmafmctl Device failover -j FilesetName
            --new-target NewAfmTarget [--target-only] [-s LocalWorkDirectory]
```

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 in the NSD protocol the mount point on the gateway nodes of the **afmTarget** filesets does not match the mount point on the 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:

```
mmafmctl Device evict -j FilesetName
            [--safe-limit SafeLimit] [--order {LRU | SIZE}]
            [--log-file LogFile] [--filter Attribute=Value ...]
            [--list-file ListFile] [--file FilePath]
```

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.

--list-file *ListFile*
: Contains a list of files that you want to evict, one file per line. All files must have fully qualified path names. Filesystem quotas need not be specified. 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.

--file *FilePath*
: The fully qualified name of the file that needs to be evicted. Filesystem quotas need not be specified.

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:

```
mmafmctl 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.

--rpo *RPO*
Specifies the RPO interval in minutes for this primary fileset.

10. This section describes:

```
mmafmctl 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:

```
mmafmctl 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:

```
mmafmctl 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:

mmafmctl *Device* **failbackToPrimary -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. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*..

Examples

1. **running resync on SW:**

```
# mmafmctl fs1 resync -j sw1
mmafmctl: Performing resync of fileset: sw1

# mmafmctl fs1 getstate -j sw1
```

```
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:**

```
# mmafmctl fs1 expire -j ro1

# mmafmctl fs1 getstate -j ro1
Fileset Name Fileset Target             Cache State Gateway Node Queue Length Queue numExec
------------ --------------             ----------- ------------ ------------ -------------
ro1          gpfs:///gpfs/remotefs1/dir1 Expired     c26c4apv1    0            4
```

3. **Unexpiring a RO fileset:**

```
# mmafmctl fs1 unexpire -j ro1

# mmafmctl fs1 getstate -j ro1
Fileset Name Fileset Target             Cache State Gateway Node Queue Length Queue numExec
------------ --------------             ----------- ------------ ------------ -------------
ro1          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
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 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            3
```

13. Manual evict using the *--list-file* option:

```
[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCache
total 6.0M
27858308 1.0M -rw-r--r--. 1 root root 1.0M Feb  5 02:07 file1M
27858307 2.0M -rw-r--r--. 1 root root 2.0M Feb  5 02:07 file2M
27858306 3.0M -rw-r--r--. 1 root root 3.0M Feb  5 02:07 file3M

[root@c21f2n08 ~]# echo "RULE EXTERNAL LIST 'HomePREPDAEMON' RULE 'ListLargeFiles'
 LIST 'HomePREPDAEMON' WHERE PATH_NAME LIKE '%'" > /tmp/evictionPolicy.pol


[root@c21f2n08 ~]# mmapplypolicy /gpfs/fs1/evictCache -I defer -P /tmp/evictionPolicy.pol -f /tmp/evictionList

#Edited list of files to be evicted
[root@c21f2n08 ~]# cat /tmp/evictionList.list.HomePREPDAEMON
27858306 605742886 0   -- /gpfs/fs1/evictCache/file3M

[root@c21f2n08 ~]# mmafmctl fs1 evict -j evictCache --list-file /tmp/evictionList.list.HomePREPDAEMON

[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCache
total 3.0M
27858308 1.0M -rw-r--r--. 1 root root 1.0M Feb  5 02:07 file1M
27858307 2.0M -rw-r--r--. 1 root root 2.0M Feb  5 02:07 file2M
27858306    0 -rw-r--r--. 1 root root 3.0M Feb  5 02:07 file3M
```

14. Manual evict using the *--file* option:

```
[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCache
total 3.0M
27858308 1.0M -rw-r--r--. 1 root root 1.0M Feb  5 02:07 file1M
27858307 2.0M -rw-r--r--. 1 root root 2.0M Feb  5 02:07 file2M
27858306    0 -rw-r--r--. 1 root root 3.0M Feb  5 02:07 file3M

[root@c21f2n08 ~]# mmafmctl fs1 evict -j evictCache  --file /gpfs/fs1/evictCache/file1M

[root@c21f2n08 ~]# ls -lshi /gpfs/fs1/evictCache
total 0
27858308 0 -rw-r--r--. 1 root root 1.0M Feb  5 02:07 file1M
27858307 0 -rw-r--r--. 1 root root 2.0M Feb  5 02:07 file2M
27858306 0 -rw-r--r--. 1 root root 3.0M Feb  5 02:07 file3M
```

See also

- "mmafmconfig command" on page 37
- "mmafmlocal command" on page 54
- "mmchattr command" on page 113
- "mmchconfig command" on page 123
- "mmchfileset command" on page 163
- "mmchfs command" on page 169
- "mmcrfileset command" on page 223
- "mmcrfs command" on page 229
- "mmlsconfig command" on page 359

- "mmlsfileset command" on page 365
- "mmlsfs command" on page 369
- "mmpsnap command" on page 447

See the AFM and AFM-based DR chapters in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* for details.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To list information of all the cached files in a fileset:

```
mmafmlocal ls
```

   The system displays information similar to:

```
total 10240
-rwxrwxrwx 1 root root 10485760 May 24 09:20 file1
```

2. To list information of a specific cached file in a fileset:

```
mmafmlocal ls file2
```

   The system displays information similar to:

```
-rwxrwxrwx 1 root root 10485760 May 24 09:20 file2
```

3. To list the file statistics:

```
mmafmlocal stat file2
```

The system displays information similar to:

```
File: file2
   Inode number:  1582477
   Device ID:     0x2C (44)
   Size:          10485760
   Blocks:        20480
   Block size:    262144
   Links:         1
   Uid:           0 (root)
   Gid:           0 (root)
   Mode:          0100777
   Access time:   1464281093 (Thu May 26 16:44:53 2016 UTC)
   Modify time:   1464096038 (Tue May 24 13:20:38 2016 UTC)
   Change time:   1464096038 (Tue May 24 13:20:38 2016 UTC)
```

See also

- "mmafmconfig command" on page 37
- "mmafmctl command" on page 40
- "mmchattr command" on page 113
- "mmchconfig command" on page 123
- "mmchfileset command" on page 163
- "mmchfs command" on page 169
- "mmcrfileset command" on page 223
- "mmcrfs command" on page 229
- "mmlsconfig command" on page 359
- "mmlsfileset command" on page 365
- "mmlsfs command" on page 369
- "mmpsnap command" on page 447

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 MaxFiles] [--max-sort-bytes MaxBytes]
              [--other-sort-options SortOptions] [--qos QosClass]
              [--scope {filesystem | fileset | inodespace}]
              [--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 Guide*.

For more information about policy rules, see the topic *Policies for automating file management* in the *IBM Spectrum Scale: 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: 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 ?        Sl   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    Sl   07:29   0:38 /usr/lpp/mmfs/bin/tsapolicy /mak/millions -P /ghome/makaplan/policies/lp.policy -I yes -L 1 -X 10.222.4.12 -N all
root      3850  0.0  0.0  84832  1620 pts/1    Sl   07:29   0:00 /usr/lpp/mmfs/bin/tsapolicy /mak/millions -P /ghome/makaplan/policies/lp.policy -I 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 *IscanBuckets*
: 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 *IscanThreads*
: 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.

  For more information about file list records, refer to the *IBM Spectrum Scale: 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: 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" on page 58.

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.

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: 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 {all | 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 Guide*.

-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: 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.

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 **LIMIT**s 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 *MaxFiles*
Specifies the maximum number of files to be passed as input to the **sort** command for sorting and merging.

The **mmapplypolicy** command must do multiple, potentially large sorts and merges of temporary files as part of its processing. The **--max-merge-files** parameter specifies the maximum number of files that the **mmapplypolicy** command passes as input to a single **sort** command for sorting and merging.

If you set this value too high, the result can be excessive memory usage. The operating system can respond by terminating some of the **sort** processes, which causes the **mmapplypolicy** command to run for a longer time or to return with an error.

The default value of this parameter is 12. In general, it is a good practice to accept the default value of this parameter or to test carefully if you specify an overriding value.

See the related parameters **--max-sort-bytes** and **--sort-buffer-size**.

--max-sort-bytes *MaxBytes*
Specifies the maximum number of bytes to be passed as input files to the **sort** command. This parameter does not apply to merges in which each of the input files has already been sorted.

The **mmapplypolicy** command must do multiple, potentially large sorts and merges of temporary files as part of its processing. The **--max-sort-bytes** parameter specifies the maximum number of bytes that the **mmapplypolicy** command can pass to a single instance of the **sort** command in one or more files.

If you set this value too high, the result can be excessive memory usage. The operating system can respond by terminating some of the **sort** processes, which causes the **mmapplypolicy** command to run for a longer time or to return with an error.

The default value of this parameter is 411MB. In general, it is a good practice to accept the default value of this parameter or to test carefully if you specify an overriding value.

See the related parameters **--max-merge-files** and **--sort-buffer-size**.

--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.

fileset
The scope of the scan is limited to the objects in the same fileset as the directory pointed to by the *Directory* parameter.

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.

--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" on page 196. 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: 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*
Sets the sort-buffer size that is passed to the **sort** command. This parameter limits memory usage by the **sort** commands that the **mmapplypolicy** command calls to do sorts and merges.

The **mmapplypolicy** command must do multiple, potentially large sorts and merges of temporary files as part of its processing. It calls the operating-system **sort** command each time that it must do a sort. It can have multiple instances of the **sort** command running at the same time. If the numbers of items to be sorted are very large, the result can be excessive memory usage. The operating system can respond by terminating some of the sort processes, which causes the **mmapplypolicy** command to run for a longer time or to return with an error.

To prevent excessive memory consumption, you can set the **--sort-buffer-size** parameter to a lower value than its default. The **--sort-buffer-size** parameter is the value that the **mmapplypolicy** command passes to the **sort** command in the **buffer-size** parameter. The default value is 8%. If you need a lower value, you might set it to 5%.

You can specify the **--sort-buffer-size** parameter in any format that the **sort** program's **buffer-size** parameter accepts, such as "5%" or "1M".

In general, accept the default value of this parameter unless the system has excessive memory consumption that is attributable to large sort operations by the **mmapplypolicy** command.

See the related parameters **--max-merge-files** and **max-sort-bytes**.

--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 *Improving performance with the --sort-command parameter* in *IBM Spectrum Scale: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 fs1 -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%
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: Administration Guide*.

See also

- "mmchpolicy command" on page 191
- "mmcrsnapshot command" on page 246
- "mmlspolicy command" on page 384
- "mmlssnapshot command" on page 395
- "mmsnapdir command" on page 511

Location

`/usr/lpp/mmfs/bin`

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 [-l CipherList]
```

or

```
mmauth update RemoteClusterName {[-C NewClusterName] [-k KeyFile] [-l 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 **mmremotecluster** 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 **mmremotecluster 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 *Specifying nodes as input to GPFS(tm) commands* in *IBM Spectrum Scale: Administration Guide*.

This command does not support a *NodeClass* of **mount**.

RemoteClusterName
Specifies the remote cluster name requesting access to local GPFS file systems.

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: 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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: 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 SHA digest* in *IBM Spectrum Scale: Problem Determination Guide*.

See also

- "mmremotefs command" on page 460
- "mmremotecluster command" on page 457

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: Administration Guide*.

Location

`/usr/lpp/mmfs/bin`

mmbackup command

Performs a backup of a GPFS file system or independent fileset to an IBM Spectrum Protect 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: Administration Guide*.

The IBM Spectrum Protect 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: Administration Guide*.

You can run multiple instances of **mmbackup**, as long as they are on different file systems.

If you are planning to use IBM Spectrum Protect to back up IBM Spectrum Scale file systems, see the topic *Backup considerations for using IBM Spectrum Protect* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the topic *Configuration reference for using IBM Spectrum Protect with IBM Spectrum Scale* in the *IBM Spectrum Scale: Administration Guide*.

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.

-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 IBM Spectrum Protect 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 Guide*.

-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 **mmlssnapshot** 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 76.

-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" on page 196. 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: Administration Guide*.

--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 IBM Spectrum Protect 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 IBM Spectrum Protect 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 **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.

-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 IBM Spectrum Protect Backup-Archive client program dsmc

MMBACKUP_DSMC_MISC
The value of this variable is passed as arguments to **dsmc restore** and **dsmc query** {**backup**,**inclexcl**,**session**} commands.

MMBACKUP_DSMC_BACKUP
The value of this variable is passed as arguments to **dsmc**, **dsmc selective**, and **dsmc incremental** commands.

MMBACKUP_DSMC_EXPIRE
The value of this variable is passed as arguments to **dsmc 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: 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.

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 IBM Spectrum Protect 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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.
-------------------------------------------------------
```

2. 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.
   -------------------------------------------------------
   ```

3. 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: 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 56

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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 329
- "mmimgrestore command" on page 333
- "mmrestoreconfig command" on page 467

Location

```
/usr/lpp/mmfs/bin
```

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 similar to:

```
Wrote: /root/rpmbuild/RPMS/x86_64/gpfs.gplbin-3.10.0-229.el7.x86_64-4.2.1-0.x86_64.rpm
```

or

```
Wrote: /tmp/deb/gpfs.gplbin_4.2.1-0_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* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Location

/usr/lpp/mmfs/bin

mmcallhome command

Manages the call home operations.

Synopsis

mmcallhome group add *groupName server* [--node {all | childNode [,*childNode*...]}]

or

mmcallhome group list

or

mmcallhome group delete *GroupName*

or

mmcallhome group auto [[**--server** {*all* | {*ServerName*[,*ServerName*...]}] | **--force** | {**--enable** [*LICENSE* | *ACCEPT*] | **--disable**] }]

or

mmcallhome capability list

or

mmcallhome capability enable

or

mmcallhome capability disable

or

mmcallhome info list

or

mmcallhome info change [**--customer-name** *CustomerName* | **--customer-id** *CustomerId* | **--email** *Email* | **--country-code** *CountryCode*

or

mmcallhome proxy enable [**--with-proxy-auth**]

or

mmcallhome proxy disable

or

mmcallhome proxy list

or

mmcallhome proxy change [**--proxy-location** *ProxyLocation* | **--proxy-port** *ProxyPort* | **--proxy-username** *ProxyUsername* | **--proxy-password** *ProxyPassword*]

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 the concept of the 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 initiates the data collection from the call home child nodes and uploads data to IBM using the HTTPS protocol. 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 call home feature, see *Monitoring the IBM Spectrum Scale(tm) system remotely by using call home* in *IBM Spectrum Scale: Problem Determination 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.

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.

--force
Creates new groups after deleting the existing groups.

Note: If this option is defined and no server node has been specified, potential server nodes get detected automatically. If no server gets detected, an error is returned and the existing groups does not get deleted.

--enable
Enables the cluster for call home functionality. This parameter allows creation of new groups, if needed. The **enable** parameter, by default shows the license and asks for acceptance, if no other option is defined.

license
Shows license and terminate.

accept
Does not show license and assumes that the license is accepted.

Note: It is not possible to use this option multiple times or together with the −−disable option.

--disable
Disable call home.

Note: All groups are disabled.

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 9. key-value

Key	Value
customer-name	Business/company name This name can consist of any alphanumeric characters and these non-alphanumeric characters: '-', '_', '.', ' ', ','
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 10. 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.

--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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
callhome-method                ethernet
country-code                   JP
```

5. To create a call home group automatically, issue this command:

```
mmcallhome group auto
```

   The system displays output similar to this:

```
mmcallhome group auto: [I] In progress: Create 1 new call home groups.
mmcallhome group auto: [I] In progress: Nodes without call home: 1 See /var/adm/ras/mmfs.log.latest for details.
group: autoGroup_1 successfully added
```

6. To create a call home group automatically and enable the cluster for call home functionality by displaying options for acceptance, issue this command:

```
mmcallhome group auto --enable
```

   The system displays output similar to this:

```
By accepting this request, you agree to allow IBM and its subsidiaries to store and use your contact
information and your support information anywhere they do business worldwide. For more information, please refer
 to the Program license agreement and documentation. If you agree, please respond with "accept" for acceptance,
else with "not accepted" to decline.
(accept / not accepted)
accept
mmcallhome group auto: [I] In progress: Create 1 new call home groups.
mmcallhome group auto: [I] In progress: Nodes without call home: 1 See /var/adm/ras/mmfs.log.latest for details.
group: autoGroup_1 successfully added
```

   Note: To accept the call home functionality, type **accept** manually. Type `mmcallhome group auto --enable accept` to avoid the explicit acceptance from the user.

7. To use create new group after deleting the existing group, issue this command:

```
mmcallhome group auto --force
```

The system displays output similar to this:

```
mmcallhome group auto: [I] In progress: Create 1 new call home groups.
mmcallhome group auto: [I] In progress: Nodes without call home: 1 See /var/adm/ras/mmfs.log.latest for details.
mmcallhome group auto: [I] In progress: Delete existing groups.
Call home group autoGroup_1 has been deleted
group: autoGroup_2 successfully added
```

8. 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
```

9. 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/gpfscallhome_gatherSend_daily.conf registered
41 command entries are defined for this task
```

10. 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/gpfscallhome_gatherSend_weekly.conf registered
14 command entries are defined for this task
```

11. 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
```

Note: The CronParameter indicates the date and time settings for the execution of the command. It displays the values for minutes (0-59), hours (0-23), day of month (1-31), month (1-12 or jan-dec) and day of week (0-6, where sun=0 or sun-sat). For example `CronParameter 54 3 * * sun` indicates that the command executes on every Sunday at 3.54 am.

12. 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.cl0.DC
daily  20150930133134.802  success       ...aultDaily.g_daily.20150930133134802.cl0.DC
daily  20150930133537.509  success       ...aultDaily.g_daily.20150930133537509.cl0.DC
daily  20150930133923.063  success       ...aultDaily.g_daily.20150930133923063.cl0.DC
RunSendFile 20150930133422.843 success
...group2.MyTestData.s_file.20150930133422843.cl0.DC
```

13. 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
```

14. 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
```

15. 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
```

16. 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
```

17. 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 ]
-------------------------------------------------------------------------------
               31790849425327.4_2_1_0.x.abc.autoGroup_1.gat_weekly.g_weekly.20160412160447854.cl0.DC
-------------------------------------------------------------------------------
               31790849425327.4_2_1_0.x.abc.autoGroup_1.gat_weekly.g_weekly.20160412173941161.cl0.DC
-------------------------------------------------------------------------------
weekly         20160412174030.803  20160412174034        failed       RC=6 (lock err)
               NA
-------------------------------------------------------------------------------
               31790849425327.4_2_1_0.x.abc.autoGroup_1.gat_weekly.g_weekly.20160412175159390.cl0.DC
```

Note: Sometimes the output of `mmcallhome status list --verbose` displays single line without detailed information about RC indicating successful completion of call home tasks. The failed status indicates that there was an issue with the call home task and the RC numeral indicates the respective issue. If the value of RC is zero, it indicates that the upload procedure is successful, but some automatically resolvable issue occurred while uploading the data. The value, RC != 0, indicates that the upload procedure is not successful. The detailed information about the upload procedure is available in the logs.

18. 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
```

19. 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 123
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359
- "mmnfs command" on page 402
- "mmobj command" on page 414
- "mmsmb command" on page 500
- "mmuserauth command" on page 527

Location

```
/usr/lpp/mmfs/bin
```

mmces command

Manage CES (Cluster Export Services) configuration.

Synopsis

```
mmces address add [--ces-node Node] [--attribute Attribue] [--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:

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 reconfigures its network interfaces to no longer accept communication for that 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 command associates the attributes that are specified by the **--attribute** parameter with the address that is specified by the **--ces-ip** parameter. If an attribute is already associated with another address, that association is ended.
- If the **--remove-attribute** parameter is specified, the command removes the attributes that are specified by the **--attribute** parameter from the addresses that are specified by the **--ces-ip** parameter.
- The command associates the groups that are specified by the **--ces-group** parameter with the address that is specified by the **--ces-ip** parameter
- If the **--remove-group** parameter is specified, the command removes the groups that are specified by the **--ces-group** parameter from the addresses that are specified by the **--ces-ip** parameter.

If the **--ces-ip** parameter is not specified:

- If the **--remove-attribute** parameter is specified, the command removes the attributes that are specified by the **--attribute** parameter from their current associations.
- If the **--remove-group** parameter is specified, the command removes the groups that are specified by the **--group** parameter from their current associations.
- Specifying **--remove-group** with the groups specified by the **--group** parameter removes the groups from their current associations.

list
Lists the CES addresses along with group, attribute and node assignments.

Options:

--ces-ip List only the addresses provided.

--ces-group List only addresses whose group assignment matches one of the groups provided.

--attribute List only addresses whose attributes match one of the attributes provided.

--ces-node List only 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.

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 Guide*.

-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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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.

6. 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.

7. 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.

8. 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.

9. 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.
```

10. To resume the current node, issue this command:
```
mmces node resume
```

The system displays output similar to this:
```
Node no longer in suspended state.
```

11. 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.

12. 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.

13. 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.

14. 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.

15. 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
```

16. To display the current CES log level, issue this command:
```
mmces log level
```

The system displays output similar to this:

```
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 123
- “mmlscluster command” on page 357
- “mmlsconfig command” on page 359
- “mmnfs command” on page 402
- “mmobj command” on page 414
- “mmsmb command” on page 500
- “mmuserauth command” on page 527

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 by 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 by 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 cluster disaster recovery* in *IBM Spectrum Scale: Administration Guide*.

The **mmcesdr** log file is at `/var/adm/ras/mmcesdr.log`. This log file is included with the CES information generated by the **gpfs.snap** command. The **gpfs.snap** command generates the CES information by default, if a protocol is enabled.

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 <= RPO <= 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 that is saved in the configuration independent fileset is used as default.

--file-config {--recreate | --restore}

Optional. Specifies whether SMB and NFS exports are re-created, or if the entire protocol configuration is restored. If not specified, the SMB and NFS exports are 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 is 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. This step specifically converts the newly created independent filesets to primary AFM DR filesets.

--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 was 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 is 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 is 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 that is saved in the configuration independent fileset is used as default.

--file-config {--recreate | --restore}

Optional. Specifies whether SMB and NFS exports are re-created, or if the entire protocol configuration is restored. If not specified, the SMB and NFS exports are 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 that is 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 are re-created, or if the entire protocol configuration is restored. If not specified, the SMB and NFS exports are re-created by default.

Exit status

0 Successful completion.

nonzero
A failure has occurred.

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. For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*..

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 are 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.

   Note: The outband data transfer is the default method of data transfer from the primary cluster to the secondary cluster when AFM DR fileset relationships are first set up.
   :

```
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 outband 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-ganesha1 to fileset fs0:nfs-ganesha1 on secondary cluster.
Transfer all data on primary cluster for fileset fs0:nfs-ganesha2 to fileset fs0:nfs-ganesha2 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 outband 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 setup 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 is 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-ganesha1/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-3 to fileset link point of fileset fs0:nfs-ganesha1 on new primary cluster.
   Transfer all data under snapshot located on acting primary cluster at:
   /gpfs/fs0/nfs-ganesha2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-4 to fileset link point of fileset fs0:nfs-ganesha2 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
```

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 96
- “mmchconfig command” on page 123
- “mmlscluster command” on page 357
- “mmlsconfig command” on page 359
- “mmnfs command” on page 402
- “mmsmb command” on page 500
- “mmobj command” on page 414
- “mmuserauth command” on page 527

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 **mmlsattr** 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 RAIDs 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.

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 Guide*.

--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 *FailureGroupx* 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

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:

   ```
   mmlsattr 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
   ```

See also

- "mmcrfs command" on page 229
- "mmlsattr command" on page 352
- "mmlsfs command" on page 369

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: 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.

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: 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: 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**, **mmremotecluster**, 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 Guide*.

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:

```
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/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
```

See also

- "mmaddnode command" on page 29
- "mmchnode command" on page 180
- "mmcrcluster command" on page 218
- "mmdelnode command" on page 276
- "mmlscluster command" on page 357
- "mmremotecluster command" on page 457

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**
- **deadlockOverloadThreshold**
- **dmapiEventTimeout**
- **dmapiMountTimeout**
- **dmapiSessionFailureTimeout**
- **expelDataCollectionDailyLimit**
- **expelDataCollectionMinInterval**
- **fastestPolicyCmpThreshold**
- **fastestPolicyMaxValidPeriod**
- **fastestPolicyMinDiffPercent**
- **fastestPolicyNumReadSamples**
- **fileHeatLossPercent**
- **fileHeatPeriodMinutes**
- **ignorePrefetchLUNCount**
- **lrocData**
- **lrocDataMaxFileSize**
- **lrocDataStubFileSize**

- **lrocDirectories**
- **lrocInodes**
- **maxMBpS**
- **nfsPrefetchStrategy**
- **nsdBufSpace**
- **nsdInlineWriteMax**
- **nsdMultiQueue**
- **pagepool**
- **pitWorkerThreadsPerNode**
- **readReplicaPolicy**
- **seqDiscardThreshold**
- **syncbuffsperiteration**
- **systemLogLevel**
- **unmountOnDiskFail**
- **verbsRdmaRoCEToS**
- **verbsRdmasPerConnection**
- **verbsRdmasPerNode**
- **verbsSendBufferMemoryMB**
- **worker1Threads** (only when adjusting value down)
- **writebehindThreshold**

-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**
- **commandAudit**
- **dataDiskWaitTimeForRecovery**
- **dataStructureDump**
- **deadlockBreakupDelay**
- **deadlockDataCollectionDailyLimit**
- **deadlockDataCollectionMinInterval**
- **deadlockDetectionThreshold**
- **deadlockDetectionThresholdForShortWaiters**
- **deadlockOverloadThreshold**
- **debugDataControl**
- **disableInodeUpdateOnFDatasync**
- **dmapiEventTimeout**
- **dmapiMountTimeout**
- **dmapiSessionFailureTimeout**
- **expelDataCollectionDailyLimit**
- **expelDataCollectionMinInterval**

- **fastestPolicyCmpThreshold**
- **fastestPolicyMaxValidPeriod**
- **fastestPolicyMinDiffPercent**
- **fastestPolicyNumReadSamples**
- **fileHeatLossPercent**
- **fileHeatPeriodMinutes**
- **forceLogWriteOnFdatasync**
- **ignorePrefetchLUNCount**
- **lrocData**
- **lrocDataMaxFileSize**
- **lrocDataStubFileSize**
- **lrocDirectories**
- **lrocInodes**
- **maxDownDisksForRecovery**
- **maxFailedNodesForRecovery**
- **maxMBpS**
- **metadataDiskWaitTimeForRecovery**
- **minDiskWaitTimeForRecovery**
- **nfsPrefetchStrategy**
- **nsdBufSpace**
- **nsdInlineWriteMax**
- **nsdMultiQueue**
- **pagepool**
- **pitWorkerThreadsPerNode**
- **readReplicaPolicy**
- **restripeOnDiskFailure**
- **seqDiscardThreshold**
- **syncbuffsperiteration**
- **systemLogLevel**
- **unmountOnDiskFail**
- **verbsRdmaRoCEToS**
- **verbsRdmasPerConnection**
- **verbsRdmasPerNode**
- **verbsSendBufferMemoryMB**
- **worker1Threads** (only when adjusting value down)
- **writebehindThreshold**

-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: Administration Guide*.

To see a complete list of the attributes for which the **-N** flag is valid, see the see the table "Configuration attributes on the **mmchconfig** command" in the topic *Changing the GPFS cluster configuration data* in the *IBM Spectrum Scale: 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:

allToAll
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.

For more information, see the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

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 disable.

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 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 *Peer snapshot -psnap* in *IBM Spectrum Scale: Concepts, Planning, and Installation 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: 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. For the CES shared root, the recommended value is a dedicated file system, but it is not enforced. The CES shared root can also be a part of an existing GPFS™ file system. In any case, **cesSharedRoot** must reside on GPFS and must be available when it is configured through **mmchconfig**.

GPFS must be down on all CES nodes in the cluster when changing the **cesSharedRoot** attribute.

cifsBypassTraversalChecking
Controls the GPFS behavior while performing access checks for directories

GPFS grants the SEARCH access when the following conditions are met:

- The object is a directory
- The parameter value is **yes**
- The calling process is a Samba process

GPFS grants the SEARCH access regardless of the mode or ACL.

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: 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: 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: 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: Administration Guide* for additional information.

commandAudit
Controls the logging of audit messages for GPFS commands that change the configuration of the cluster. This attribute is not supported on Windows operating systems. For more information, see the topic *Audit messages for cluster configuration changes* in the *IBM Spectrum Scale: Problem Determination Guide*.

on Starts audit messages. Messages go to syslog and the GPFS log.

syslogOnly
Starts audit messages. Messages go to syslog only. This value is the default.

off Stops audit messages.

The **-N** flag is valid for this attribute.

dataDiskCacheProtectionMethod
The **dataDiskCacheProtectionMethod** parameter defines the cache protection method for disks that are used for the GPFS file system. The valid values for this parameter are 0, 1, and 2.

The default value is 0. The default value indicates that the disks are Power-Protected and, when the down disk is started, only the standard GPFS log recovery is required. If the value of this parameter is 1, the disks are Power-Protected with no disk cache. GPFS works the same as before. If the value of this parameter is 2, when a node stops functioning, files that have data in disk cache must be recovered to a consistent state when the disk is started.

This parameter impacts only disks in the FPO storage pool. If the physical disk-write cache is enabled, the value of this parameter must be set to 2. Otherwise, maintain the default.

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 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.

dataStructureDump
Specifies a path for storing dumps. You can specify a directory or a symbolic link. The default is to store dumps in `/tmp/mmfs`. This attribute takes effect immediately whether or not `-i` is specified.

It is a good idea to create a directory or a symbolic link for problem determination information. Do not put it in a GPFS file system, because it might not be available if GPFS fails. When a problem occurs, GPFS can write 200 MB or more of problem determination data into the directory. Copy and delete the files promptly so that you do not get a **NOSPACE** error if another failure occurs.

Important: Before you change the value of **dataStructureDump**, stop the GPFS trace. Otherwise you will lose GPFS trace data. Restart the GPFS trace afterwards. For more information, see the topic *Generating GPFS trace reports* in the *IBM Spectrum Scale: Problem Determination Guide*.

The **-N** flag is valid for this attribute.

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 each day.

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 3600 seconds.

deadlockDetectionThreshold
Specifies the initial deadlock detection threshold. The effective deadlock detection threshold adjusts itself over time. A suspected deadlock is detected when a waiter waits longer than the effective deadlock detection threshold.

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.

deadlockOverloadThreshold
Specifies the threshold for detecting a cluster overload condition. If the overload index on a node exceeds the deadlockOverloadThreshold, then the effective deadlockDetectionThreshold is raised. The overload index is calculated heuristically and is based mainly on the I/O completion times.

The default is 1. If the value is 0, then overload detection is disabled.

debugDataControl
Controls the amount of debug data that is collected. This attribute takes effect immediately whether or not `-i` is specified. The `-N` flag is valid for this attribute.

none No debug data is collected.

light The minimum amount of debug data that is most important for debugging issues is collected. This is the default value.

medium
More debug data is collected.

heavy The maximum amount of debug data is collected, targeting internal test systems.

verbose
Needed only for troubleshooting special cases and can result in very large dumps.

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 Guide*.

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 *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

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 *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

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 *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

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 *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

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 *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

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 3. 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 3600 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. In a system with SSD and regular disks, the value of the **fastestPolicyCmpThreshold** parameter can be set to a greater number to let GPFS refresh the speed statistics for slower disks less frequently.

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, see *Encryption* in *IBM Spectrum Scale: 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.

When **forceLogWriteOnFdatasync** is set to **no**, the GPFS log record is flushed only when a new block is written to the file.

ignorePrefetchLUNCount
The GPFS client node calculates the number of sequential access prefetch and write-behind threads to run concurrently for each file system by using the count of the number of LUNs in the file system and the value of **maxMBpS**. However, if the LUNs being used are composed of multiple physical disks, this calculation can underestimate the amount of IO that can be done concurrently.

Setting the value of the **ignorePrefetchLUNCount** parameter to yes does not include the LUN count and uses the **maxMBpS** value to dynamically determine the number of threads to schedule the **prefetchThreads** value.

This parameter impacts only the GPFS client node. The GPFS NSD server does not include this parameter in the calculation.

The valid values for this parameter are yes and no. The default value is yes and can be used in traditional LUNs where one LUN maps to a single disk or an n+mP array. Set the value of this parameter to yes when the LUNs presented to GPFS are made up of a large numbers of physical disks.

The **-N** flag is valid for this attribute.

lrocData
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 **lrocData** 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 **lrocDataMaxFileSize** and **lrocDataStubFileSize** configuration options to save all data from small files or all data from the initial portion of large files.

lrocDataMaxFileSize
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.

lrocDataStubFileSize
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.

lrocDirectories
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**.

lrocInodes
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.

maxBufferDescs
Valid values are from 512 to 10,000,000.

Without explicit setting, it is set to a value of 10 * **maxFilesToCache** up to pagepool size/16KB. Each buffer descriptor caches maximum block size data for a file. When caching small files, it does not actually need to be more than a small multiple of **maxFilesToCache** since only OpenFile objects can cache data blocks. When an application needs to cache very large files, **maxBufferDescs** can be tuned to ensure that there are enough to cache large files.

For example, if you have 10,000 buffer descriptors configured and a 1MiB file system blocksize, you will not have enough buffer descriptors to cache a 20GiB file. To cache a 20GiB file, increase **maxBufferDescs** to at least 20,480 (20GiB/1MiB=20,480).

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.

maxMissedPingTimeout
See the **minMissedPingTimeout** parameter.

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***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.

minMissedPingTimeout
The **minMissedPingTimeout** and **maxMissedPingTimeout** parameters set limits on the calculation of **missedPingTimeout** (MPT). The MPT is the allowable time for pings sent from the Cluster Manager (CM) to a node that has not renewed its lease to fail. The default MPT value is 5 seconds less than **leaseRecoveryWait**. The CM will wait the MPT seconds after the lease has expired before declaring a node out of the cluster. The values of the **minMissedPingTimeout** and **maxMissedPingTimeout** are in seconds; the default values are 3 and 60 respectively. If these values are changed, only GPFS on the quorum nodes that elect the CM must be recycled to take effect.

This parameter can be used to cover over a central network switch failure timeout or other network glitches that might be longer than **leaseRecoveryWait**. This might prevent false node down conditions, but it will extend the time for node recovery to finish and may block other nodes from progressing if the failing node holds the tokens for many shared files.

As is the case with **leaseRecoveryWait**, a node is usually expelled from the cluster if there is a problem with the network or the node runs out of resources like paging. For example, if there is an application running on a node that is paging the machine too much or overrunning network capacity, GPFS might not have the chance to contact the Cluster Manager node to renew the lease within the timeout period.

The default value of this parameter is 3. A valid value is any number ranging from 1 to 300.

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 allows more lenient intranode locking, but imposes internode whole file range tokens on files using **mmap** while writing.

nfsPrefetchStrategy
With the **nfsPrefetchStrategy** parameter, GPFS optimizes prefetching for NFS file-style access patterns. This parameter defines a window of the number of blocks around the current position that are treated as fuzzy-sequential access. The value of this parameter can improve the performance while reading large files sequentially. However, because of kernel scheduling, some read requests that come to GPFS are not sequential. If the file system blocksize is smaller than the read request sizes, increasing the value of this parameter will provide a bigger window of blocks. The default value is 0. A valid value is any number ranging are from 0 to 10.

Setting the value of **nfsPrefetchStrategy** to 1 or greater can improve the sequential read performance when large files are accessed by using NFS and the filesystem block size is smaller than the NFS transfer block size.

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 IBM Spectrum Scale 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 IBM Spectrum Scale RAID, see *IBM Spectrum Scale RAID: Administration*.

The **-N** flag is valid for this attribute.

nsdInlineWriteMax
The **nsdInlineWriteMax** parameter specifies the maximum transaction size that can be sent as embedded data in an NSD-write RPC.

In most cases, the NSD-write RPC exchange performs the following steps:

1. An RPC is sent from the client to the server to request a write.
2. A GetData RPC is sent back from the server to the client to request the data.

Note: For data smaller than **nsdInlineWriteMax**, GPFS sends that amount of write data directly without the GetData RPC from the server to the client.

The default value of this parameter is 1024. A valid value is any number ranging from 0 to 8M.

nsdMaxWorkerThreads
The **nsdMaxWorkerThreads** parameter sets the maximum number of NSD threads on an NSD server that concurrently transfers data with NSD clients. The maximum value of the addition of **worker1Threads**, **prefetchThreads**, and **nsdMaxWorkerThreads** is less than 8192 on 64-bit architectures. The default value of this parameter is 64 in Release 3.4 and 512 in Release 3.5 and later. The minimum value is 8 and the maximum value is 8192. A valid value is any number ranging from 1 to 8192.

Setting this parameter to the default value can increase the **nsdMaxWorkerThreads** for large clusters. Scale this parameter with the number of LUNs and not the number of clients. This parameter manages the flow control on the network between the clients and the servers.

Important: If you set **workerThreads** to a non-default value, do not set **nsdMaxWorkerThreads**.

Check GPFS NSD Server Design and Tuning in the IBM Spectrum Scale wiki in developerWorks® for more details.

nsdMinWorkerThreads
The **nsdMinWorkerThreads** parameter is used to increase the NSD server performance by providing a large number of dedicated threads for NSD service.

The default value of this parameter is 16. A valid value is any number ranging from 1 to 8192.

Important: If you set **workerThreads** to a non-default value, do not set **nsdMinWorkerThreads**.

Check GPFS NSD Server Design and Tuning in the IBM Spectrum Scale wiki in developerWorks for more details.

nsdMultiQueue
The **nsdMultiQueue** parameter sets the number of queues. The default value of this parameter is 256. A valid value is any number ranging from 2 to 512.

Check GPFS NSD Server Design and Tuning in the IBM Spectrum Scale wiki in developerWorks for more details.

nsdRAIDTracks
This option specifies the number of tracks in the IBM Spectrum Scale RAID buffer pool, or 0 if this node does not have a IBM Spectrum Scale RAID vdisk buffer pool. This controls whether IBM Spectrum Scale RAID services are configured. For more information about IBM Spectrum Scale 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 IBM Spectrum Scale RAID vdisk buffer pool. Valid values are within the range of 10 to 90. The default is 50 when IBM Spectrum Scale RAID is configured on the node in question; 0 when it is not. For more information about IBM Spectrum Scale 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.

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.

prefetchPct
The default value of the **prefecthPct** parameter is 20% of the pagepool value. GPFS uses this as a guideline to limit the pagepool space that is to be used for prefetch and write-behind buffers for active sequential streams. If the workload is sequential with very little caching of small files or random IO, increase the value of this parameter to 60% of the pagepool value, so that each stream can have more buffers cached for prefetch and write-behind operations.

The default value of this parameter is 20. The valid value can be any number ranging from 0 to 60.

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 disk is to read replicas. By default, GPFS 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**

In a system with SSD and regular disks, the value of **fastestPolicyCmpThreshold** can be set to a greater number to let GPFS refresh the speed statistics for the slower disks less frequently. The default value is maintained for all other configurations. The default value of this parameter is default. The valid values are default, local, and fastest.

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 IBM Spectrum Scale* 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.

The command fails with an error message if the **cipherList** configuration attribute of the cluster is not set to **AUTHONLY** or higher. For more information, see the topic *Completing the migration to a new level of IBM Spectrum Scale* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

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 **mmlsfs -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 `/var/adm/ras/autorecovery.log.<timestamp>` log.

For more information on GPFS disk fail auto recovery, see Big Data best practices in the IBM Spectrum Scale wiki in developerWorks.

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.

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.

seqDiscardThreshold
With the **seqDiscardThreshold** parameter, GPFS detects a sequential read or write access pattern and specifies what has to be done with the pagepool buffer after it is consumed or flushed by write-behind threads. This is the highest performing option in a case where a very large file is read or written sequentially. The default for this value is 1MB, which means that if a file is sequentially read and is greater than 1MB, GPFS does not keep the data in cache after consumption. There are some instances where large files are reread by multiple processes such as data analytics. In some cases, you can improve the performance of these applications by increasing the value of the **seqDiscardThreshold** parameter so that it is larger than the sets of files that have to be cached. If the value of the **seqDiscardthreshold** parameter is increased, GPFS attempts to keep as much data in cache as possible for the files that are below the threshold.

The value of **seqDiscardThreshold** is file size in bytes. The default is 1MB. Increase this value if you want to cache files that are sequentially read or written and are larger than 1MB in size. Ensure that there are enough buffer descriptors to cache the file data. For more information about buffer descriptors, see the **maxBufferDescs** parameter.

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: 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: 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 th 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 *Using remote access with public and private IP addresses* in the *IBM Spectrum Scale: Administration Guide*.

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.

syncBuffsPerIteration
This parameter is used to expedite buffer flush and the rename operations done by MapReduce jobs.

The default value is 100. It should be set to 1 for the GPFS FPO cluster for Big Data applications. Keep it as the default value for all other cases.

syncSambaMetadataOps
Is used to enable and disable the syncing of metadata operations issued by the SMB server.

If set to **yes**, **fsync()** is used after each metadata operation to provide reasonable failover behavior on node failure. This ensures that the node taking over can see the metadata changes. Enabling **syncSambaMetadataOps** can affect performance due to additional sync operations.

If set to **no**, the additional sync overhead is avoided at the potential risk of losing metadata updates after a failure.

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 **notice**.

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 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: Administration Guide*.

no The **no** option is the default value. 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. The default setting for **unmountOnDiskFail** will cause file system panic/unmount if the number of failure groups with down disk is the same or more than the metadata replication.

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 `mthca0` or `mlx4_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="mlx4_0/1/7  mlx4_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="mthca0/1 mthca0/2"
```

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 for **verbsRdmasPerConnection** is 16.

verbsRdmasPerNode
Sets the maximum number of simultaneous RDMA data transfer requests allowed per node. The default value for **verbsRdmasPerNode** is 1000. The value for **verbsRdmasPerNode** is limited by the value of the **nsdMaxWorkerThreads** setting 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 (the default), a value is calculated based on the maximum number of RDMAs 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**.

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 **prefetchThreads** parameter is described in this help topic. See the Tuning Parameters article in the IBM Spectrum Scale wiki in developerWorks for descriptions of the **logBufferCount** and **worker3Threads** parameters.

Important: After you set **workerThreads** to a non-default value, avoid setting **worker1Threads**. If you do, at first only **worker1Threads** is changed. But when IBM Spectrum Scale is restarted, all corresponding variables are automatically tuned according to the value of **worker1Threads**, instead of **workerThreads**.

worker1Threads
For some categories of file I/O, this variable controls the maximum number of concurrent file I/O operations. You can increase this value to increase the I/O performance of the file system. However, increasing this variable beyond some point might begin to degrade file system performance.

Important: After you set **workerThreads** to a non-default value, avoid setting **worker1Threads**. If you do, at first only **worker1Threads** is changed. But when IBM Spectrum Scale is restarted, all corresponding variables are automatically tuned according to the value of **worker1Threads**, instead of **workerThreads**.

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.

writebehindThreshold

The **writebehindThreshold** parameter specifies the point at which GPFS starts flushing new data out of the pagepool for a file that is being written sequentially. Until the file size reaches this threshold, no write-behind is started because the full blocks are filled.

Increasing this value will defer write-behind for new larger files, which can be useful. The workload folder contains temporary files that are smaller than the value of **writebehindThreshold** and are deleted before they are flushed from cache. The default value of this parameter is 512KiB. If the value is too large, there might be too many dirty buffers that the sync thread has to flush at the next sync interval, thereby causing a surge in disk IO. Keeping the value small ensures a smooth flow of dirty data to disk.

Note: If you set new values for **afmParallelReadChunkSize**, **afmParallelReadThreshold**, **afmParallelWriteChunkSize**, and **afmParallelWriteThreshold**; you need not relink filesets for the new values to take effect.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 29
- "mmchnode command" on page 180
- "mmcrcluster command" on page 218
- "mmdelnode command" on page 276
- "mmlsconfig command" on page 359
- "mmlscluster command" on page 357

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]
           [--qos QOSClass]
```

or

```
mmchdisk Device {suspend | empty | resume | stop | start | change}
           {-d "DiskDesc[;DiskDesc...]" | -F StanzaFile}
           [-N {Node[,Node...] | NodeFile | NodeClass}]
           [--inode-criteria CriteriaFile]
           [-o InodeResultFile]
           [--qos QOSClass]
```

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 *Recoverability considerations* topic in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

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 **mmdeldisk** 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.

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: 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 **mmadddisk** command.

device=*DiskName*
The block device name of the underlying disk device. This clause is ignored by the **mmadddisk** 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 Guide*.

--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.

illCompressed
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.

illPlaced
Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

illReplicated
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
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.

--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 **other** QoS class. (Unlike other commands that have the **--qos** option, the **mmchdisk** command runs in the **other** class by default.) This parameter has no effect unless the QoS service is enabled. For more information, see the topic "mmchqos command" on page 196.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: Administration Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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      driver sector failure holds   holds status        storage
   name      type    size  group  metadata data      availability 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     driver sector failure holds   holds                                storage
name     type   size   group  metadata data  status     availability  disk id 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
   ```

   The system displays information similar to:

   ```
   disk      driver  sector failure holds    holds status              storage
   name      type     size   group  metadata data         availability 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: Administration Guide*.
- "mmadddisk command" on page 23
- "mmchnsd command" on page 188
- "mmdeldisk command" on page 266
- "mmlsdisk command" on page 361
- "mmlsnsd command" on page 381

- "mmrpldisk command" on page 485

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

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 Guide*.

-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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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)
   ```

```
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 298
- "mmfsck command" on page 304
- "mmlsquota command" on page 391
- "mmquotaon command" on page 455
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

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: 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 (Single writer, Independent writer, and Primary) in which data from cache is pushed to home.

Valid values are between 1 and 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: 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 disable.

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. 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 detailed information, see *AFM-based Asynchronous Disaster Recovery (AFM DR)* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For more information, see the topic about caching modes in the *IBM Spectrum Scale: 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.

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 *Peer snapshot -psnap* in *IBM Spectrum Scale: Concepts, Planning, and Installation 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
```

After an AFM fileset is converted to a regular fileset, the fileset cannot be changed back to an AFM fileset.

--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).

setAclOnly
Specifies that permissions can be changed using ACL commands and API only (**chmod** will not be accepted).

chmodAndSetAcl
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.

chmodAndUpdateAcl
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 *Information lifecycle management for IBM Spectrum Scale* of *IBM Spectrum Scale: Administration Guide*.

Note: If you set new values for **afmParallelReadChunkSize**, **afmParallelReadThreshold**, **afmParallelWriteChunkSize**, and **afmParallelWriteThreshold**; you need not relink filesets for the new values to take effect.

Exit status

0 Successful completion.

nonzero
A failure has occurred.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. 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.
   ```

2. 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 169
- “mmcrfileset command” on page 223
- “mmdelfileset command” on page 271
- “mmlinkfileset command” on page 350
- “mmlsfileset command” on page 365
- “mmunlinkfileset command” on page 524

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**.

-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 *GPFS-specific mount options* in the *IBM Spectrum Scale: Administration Guide*.

-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: 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: Administration Guide*.

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 IBM Spectrum Protect hierarchical storage management (HSM) before the file system can be mounted.

For further information regarding DMAPI for GPFS, see *GPFS-specific DMAPI events* in the *IBM Spectrum Scale: Command and Programming Reference*.

--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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmlsfs 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 163
- "mmcrfs command" on page 229
- "mmdelfs command" on page 274
- "mmdf command" on page 287
- "mmfsck command" on page 304
- "mmlsfs command" on page 369
- "mmrestripefs command" on page 478

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 IBM Spectrum Scale license designation, see *IBM Spectrum Scale 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. The use of any of the following components or functions of IBM Spectrum Scale Client is not authorized:

- Configuring a virtual server in the following IBM Spectrum Scale roles: Configuration Manager, Quorum node, Manager node, Network Shared Disk (NSD) Server node, Cluster Export Services node (also known as Protocol node), Advanced File Management (AFM) Gateway node, Transparent Cloud Tiering Gateway node.
- Exporting IBM Spectrum Scale data to virtual servers that do not have a valid IBM Spectrum Scale license through any application, protocol or method, including Network File System (NFS), Server Message Block (SMB), File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), Object Protocol (OpenStack Swift, Amazon S3 API).

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. Therefore, protocol nodes also require an IBM Spectrum Scale Server license.

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. The use of any of the following components or functions of IBM Spectrum Scale FPO is not authorized:

- Configuring a virtual server in the following IBM Spectrum Scale roles: Configuration Manager, Quorum node, Manager node, Cluster Export Services node (also known as Protocol node), Advanced File Management (AFM) Gateway node, Transparent Cloud Tiering Gateway node.
- Configuring a virtual server as an IBM Spectrum Scale Network Shared Disk (NSD) Server node for providing IBM Spectrum Scale data access to virtual servers that do not have a valid IBM Spectrum Scale Server or IBM Spectrum Scale FPO license entitlement.
- Exporting IBM Spectrum Scale data to virtual servers that do not have a valid IBM Spectrum Scale license through any application, protocol or method, including Network File System (NFS), Server Message Block (SMB), File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), Object Protocol (OpenStack Swift, Amazon S3 API).

The full text of the Licensing Agreement is provided with the installation media and can be found at the IBM Software license agreements website (www.ibm.com/software/sla/sladb.nsf).

--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 Guide*.

Exit status

0 Successful completion.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 373

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
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 375

Location

```
/usr/lpp/mmfs/bin
```

mmchnode command

Changes node attributes.

Synopsis

mmchnode *change-options* **-N** {*Node*[**,***Node***...**] | *NodeFile* | *NodeClass*} [**--cloud-gateway-nodeclass** *CloudGatewayNodeClass*]

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 Guide*.

[--cloud-gateway-nodeclass *CloudGatewayNodeClass***]**
This option allows for individual nodes to be enabled rather than enabling an entire node class with -N. Use this option to specify a node class you will use for Transparent Cloud Tiering management along with the -N option where you will specify individual node names. Both -N with a node list and --cloud-gateway-node with a node class will be required.

-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.

--cloud-gateway-enable
Enables one or more nodes as Transparent Cloud Tiering nodes on the cluster based on the -N option parameters.

--cloud-gateway-disable
Disables one or more Transparent Cloud Tiering nodes from the cluster based on the -N option parameters. Only disable a Transparent Cloud Tiering node if you no longer need it to migrate or recall data from the configured cloud.

--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.

--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 more information, see *Implementing a clustered NFS environment on Linux* in *IBM Spectrum Scale: 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 more information, see *Implementing a clustered NFS environment on Linux* in *IBM Spectrum Scale: 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 | --nomanager
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 more information, see *GPFS SNMP support* in *IBM Spectrum Scale: Problem Determination 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 more information, *GPFS SNMP support* in *IBM Spectrum Scale: Problem Determination 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. 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
-------------------------------------------------------------------
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
```

3. To enable all the nodes specified in the node class TCTNodeClass1 as Transparent Cloud Tiering nodes, issue this command:

```
mmchnode --cloud-gateway-enable -N TCTNodeClass1
```

The system displays output similar to this:

```
Wed May 11 12:51:37 EDT 2016: mmchnode: Processing node c350f2u18
mmchnode: Verifying media for Transparent Cloud Tiering nodes...
mmchnode: node c350f2u18 media checks passed.
```

```
Wed May 11 12:51:38 EDT 2016: mmchnode: Processing node c350f2u22.pk.labs.ibm.com
mmchnode: node c350f2u22.pok.stglabs.ibm.com media checks passed.

Wed May 11 12:51:41 EDT 2016: mmchnode: Processing node c350f2u26.pk.labs.ibm.com
mmchnode: node c350f2u26.pok.stglabs.ibm.com media checks passed.

mmchnode: Propagating the cluster configuration data to all
affected nodes.  This is an asynchronous process.
```

You can verify the Transparent Cloud Tiering nodes by issuing this command:

```
mmcloudgateway node list
```

4. To designate only a few nodes (node1 and node2) in the node class, *TCTNodeClass1*, as Transparent Cloud Tiering server nodes, issue this command:

   ```
   mmchnode --cloud-gateway-enable -N node1,node2 --cloud-gateway-nodeclass TCTNodeClass1
   ```

   Note: It only designates node1 and node2 as Transparent Cloud Tiering server nodes from the node class, *TCTNodeClass1*. Administrators can continue to use the node class for other purposes.

5. To disable all Transparent Cloud Tiering nodes from the node class, TCTNodeClass1, issue this command:

   ```
   mmchnode --cloud-gateway-disable -N TCTNodeClass1
   ```

   The system displays output similar to this:

   ```
   Thu May 12 16:10:11 EDT 2016: mmchnode: Processing node c350f2u18
   mmchnode: Verifying Transparent Cloud Tiering node c350f2u18 can be disabled...
   mmchnode: Node c350f2u18 passed disable checks.

   Thu May 12 16:10:11 EDT 2016: mmchnode: Processing node c350f2u22.pk.labs.ibm.com
   mmchnode: Verifying Transparent Cloud Tiering node c350f2u22.pk.labs.ibm.com can be disabled...
   mmchnode: Node c350f2u22.pok.stglabs.ibm.com passed disable checks.

   Thu May 12 16:10:14 EDT 2016: mmchnode: Processing node c350f2u26.pk.labs.ibm.com
   mmchnode: Verifying Transparent Cloud Tiering node c350f2u26.pk.labs.ibm.com can be disabled...
   mmchnode: Node c350f2u26.pk.labs.ibm.com passed disable checks.

   mmchnode: Propagating the cluster configuration data to all
   affected nodes.  This is an asynchronous process.
   ```

6. To disable only a few nodes (node1 and node2) from the node class, *TCTNodeClass1*, as Transparent Cloud Tiering server nodes, issue this command:

   ```
   mmchnode --cloud-gateway-disable -N node1,node2 --cloud-gateway-nodeclass TCTNodeClass1
   ```

   Note: It only disables node1 and node2 as Transparent Cloud Tiering server nodes from the node class, *TCTNodeClass1*.

See also

- "mmchconfig command" on page 123
- "mmlscluster command" on page 357

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

To display the current members of a user-defined node class called **siteA**, issue this command:

```
mmlsnodeclass 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:

```
mmlsnodeclass 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:

```
mmlsnodeclass 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:

```
mmlsnodeclass siteA
```

The system displays information similar to:

```
Node Class Name       Members
--------------------- ------------------------------------------------------------
siteA                 linuxNodes
```

See also

- "mmcrnodeclass command" on page 239
- "mmdelnodeclass command" on page 279
- "mmlsnodeclass command" on page 379

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=*NsdName*
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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmlsnsd -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 151
- “mmcrcluster command” on page 218
- “mmcrnsd command” on page 241
- “mmlsnsd command” on page 381

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: 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*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 56
- "mmlspolicy command" on page 384

Location

/usr/lpp/mmfs/bin

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 386

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=StoragePool[,QOSClass={nnnIOPS | unlimited}][,QOSClass={nnnIOPS | unlimited}] ...]
```

or

```
mmchqos Device --disable
```

Availability

Available on all IBM Spectrum Scale editions.

Description

With the **mmchqos** command, you can regulate I/O access to a specified storage pool by allocating shares of I/O operations to 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.

A third class, **misc**, is used to count the IOPS that some critical file system processes consume. You cannot assign IOPS to this class, but its count of IOPS is displayed in the output of the **mmlsqos** command.

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 equally among the nodes that have the file system mounted.
- 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 Guide*.

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 same command to **unlimited** IOPS.

When you enter multiple **mmchqos** commands for different storage pools, QoS typically records the settings for each pool and regulates the I/O consumption of each pool accordingly. However, with the **--reset** parameter, QoS discards the settings for all pools that are not specified in the same command. You can use this feature to discard the settings for any pools that you no longer want QoS to regulate and monitor.

--force
Causes QoS to accept an IOPS value lower than 100 IOPS.

Assigning less than 100 IOPS to a class is typically ineffective, because processes in that class run for an indefinitely long time. Therefore, the **mmchqos**command rejects IOPS values less than 100IOPS with an error message, unless you specify the --force option.

StoragePool
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 unspecified pools. *Unspecified pools* are storage pools that you have not specified by name in any previous **mmchqos** command.

QOSClass
The QoS class to which IOPS are allocated. You can specify one of the following classes:

maintenance
Most I/O-intensive, potentially slow-running GPFS administration commands run in this class by default. See the list of commands that support QoS in Table 11. Typically, you allocate fewer IOPS to this QoS class so that the commands that belong to it do not reduce overall file system performance.

When you start one of these commands, you can explicitly assign it to either QoS class. In certain situations, you might assign one of these commands 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.

other All other processes that use I/O run in this class by default. 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.

Some I/O-intensive, potentially slow-running GPFS administration commands run in this class by default. (Currently just one: **mmchdisk**.) See the list of commands that support QoS in Table 11. When you start one of these commands, you can explicitly assign it to either QoS class. In certain situations, you might want to assign one of these commands to the **maintenance** class so that normal processes can finish more quickly. The assignment is effective only for the instance of the command that you are starting.

The following table lists the GPFS commands that support QoS and the QoS class that the command runs in by default:

Table 11. GPFS commands that support QoS

Commands that support QoS	Default QoS class
mmadddisk	maintenance
mmapplypolicy	maintenance
mmbackup	maintenance
mmchdisk	other

Table 11. GPFS commands that support QoS (continued)

Commands that support QoS	Default QoS class
mmcheckquota	maintenance
mmdefragfs	maintenance
mmdeldisk	maintenance
mmdelfileset	maintenance
mmdelsnapshot	maintenance
mmdf	maintenance
mmfileid	maintenance
mmfsck	maintenance
mmimgbackup	maintenance
mmimgrestore	maintenance
mmlssnapshot	maintenance
mmrestripefs	maintenance
mmrpldisk	maintenance

*nnn***IOPS**
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.
  ```

  QoS divides the IOPS allocation equally among the relevant nodes.
- **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.

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 Guide*.

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 unspecified pool. By default it sets the **other** class of each unspecified 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 unspecified pool. It also allocates 576 IOPS to the **maintenance** class of the pool `mySSDs`. By default it sets the **other** classes of each unspecified 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 unspecified 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 unspecified pools and three named pools:

Storage pool	maintenance class	other class
Each unspecified 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 unspecified pool to **unlimited**:

Storage pool	maintenance class	other class
Each unspecified 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 unspecified pools, but also any named pools that are not explicitly mentioned in this command.

Storage pool	maintenance class	other class
`system`	111IOPS	**unlimited**
Each unspecified 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 unspecified pools. It assigns 222 IOPS to the **maintenance** class of each unspecified pool. And by default, it assigns **unlimited** to the **other** class of each unspecified 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 unspecified 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 388
- *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Administration Guide*.

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: 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:

```
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 246
- "mmdelsnapshot command" on page 283

Location

```
/usr/lpp/mmfs/bin
```

mmcloudgateway command

Creates and manages the cloud storage tier.

Synopsis

```
mmcloudgateway account pre-test --cloud-type {S3 |SWIFT | SWIFT-K EYSTONE | SWIFT3 | CLEVERSAFE | CLEVERSAFE-NEW}
                                          --username UserName  [ --pwd-file PasswordFile]
                                          [--cloud-url CloudURL] [--tenant-id TenantID] [--location Location]
                                          [--object-size ObjectSize] [--server-cert-path ServerCertPath]
```

or

```
mmcloudgateway account create --cloud-nodeclass CloudNodeClass --cloud-name CloudName
                              --cloud-type {S3 |SWIFT | SWIFT-KEYSTONE | SWIFT3 | CLEVERSAFE | CLEVERSAFE-NEW}
                              --username UserName  [ --pwd-file PasswordFile]
                              --enable {TRUE | FALSE} [--cloud-url CloudURL] [--tenant-id TenantID]
                              [--location Location] [--meta-location MetaLocation] [--mpu-parts-size MPUPartsSize]
                              [--server-cert-path ServerCertPath] [--enc-enable{ TRUE | FALSE }]
                              [--etag-enable {TRUE | FALSE }]
```

or

```
mmcloudgateway account test --cloud-nodeclass CloudNodeClass --cloud-name CloudName
```

or

```
mmcloudgateway account update --cloud-nodeclass CloudNodeClass --cloud-name CloudName
                               { [--pwd-file PasswordFile ] |
                               [--mpu-parts-size MPUPartsSize] |
                               [--location Location{ --cloud-type
                               {S3 |CLEVERSAFE |CLEVERSAFE-NEW}}]}
```

or

```
mmcloudgateway account delete --cloud-nodeclass CloudNodeClass --cloud-name CloudName
```

or

```
mmcloudgateway account list --cloud-nodeclass CloudNodeClass [ --cloud-name CloudName ]
```

or

```
mmcloudgateway config set --cloud-nodeclass CloudNodeClass
                          [--port Port ]
                          [--slice-size SliceSize]
                          [--migrate-threadpool-size MigrateThreadpoolSize]
                          [--recall-threadpool-size RecallThreadpoolSize]
                          [--tracing-enable {TRUE | FALSE }]
                          [--tracing-level {comp=level [,comp=level  ...]}]
                          [--audit-enable { TRUE | FALSE }] [--rotate-key]
                          [--rkm-enable { TRUE --rkm-servername RKMServerName --rkm-port RKMPort
                           --rkm-username RKMUserName
                          [--pwd-file PasswordFile] | FALSE}]
```

or

```
mmcloudgateway config unset --cloud-nodeclass CloudNodeClass
                            {[--slice-size] | [--migrate-threadpool-size ]
                            [--recall-threadpool-size ] | [ --tracing-enable ] |
                            [ --tracing-level ] | [ --audit-enable ]}
```

or

```
mmcloudgateway config list [--cloud-nodeclass CloudNodeClass]
```

or

```
mmcloudgateway filesystem create --cloud-nodeclass CloudNodeClass
                                 --file-system FileSystem --container-prefix
                                  ContainerPrefix [--override-container-name]
```

or

```
mmcloudgateway filesystem delete --cloud-nodeclass CloudNodeClass --file-system FileSystem
```

or

```
mmcloudgateway filesystem list --cloud-nodeclass CloudNodeClass
```

or

```
mmcloudgateway node list
```

or

```
mmcloudgateway service start [-N { Node [,Node ...] | NodeFile | NodeClass }]
```

or

```
mmcloudgateway service stop [-N { Node [,Node ...] | NodeFile | NodeClass }]
```

or

```
mmcloudgateway service status [-N { Node [,Node ...] | NodeFile | NodeClass }]
```

or

```
mmcloudgateway files migrate [-v] [--] File [File ...]
```

or

```
mmcloudgateway files recall [-v] [--co-resident-state]  [--] File [File ...]
```

or

```
mmcloudgateway files restore [-v] [--overwrite]
                                 { -F FileListFile | [--dry-run] [--restore-location RestoreLocation]
                                 [ --id ID] [--] File}
```

or

```
mmcloudgateway files delete {-delete-local-file | -recall-cloud-file | --require-local-file}
                            [--keep-last-cloud-file] [--] File [File ...]
```

or

```
mmcloudgateway files reconcile {Device | all}
```

or

```
mmcloudgateway files cloudList {--path Path [ --recursive  [--depth Depth]] [--file File] |
                                [--file-versions File] | --files-usage --path Path [ --depth Depth]
                                | --reconcile-status --path Path }
```

or

```
mmcloudgateway files rebuildDB Device
```

or

```
mmcloudgateway files list File [File ...]
```

Availability

Available with IBM Spectrum Scale Advanced Edition.

mmcloudgateway

Description

Use the **mmcloudgateway** command to manage and administer the Transparent Cloud Tiering feature.

Parameters

account
Manages cloud storage accounts with one of the following actions:

pre-test
Verifies that the cloud storage account settings are correct before the account is actually created. If no errors, you can go ahead and create a cloud storage.

--cloud-type CloudType
Specifies the name of the object storage provider. The following object storage providers are supported:

- Amazon Simple Storage Service (S3)
- OpenStack Swift (SWIFT)
- OpenStack Swift with Keystone authentication (SWIFT-KEYSTONE)
- S3 API layer on Swift (SWIFT3)
- IBM Cloud Object Storage version earlier than 3.7.2. To use this cloud type, select the "Cleversafe" cloud type.
- IBM Cloud Object Storage version 3.7.2 and above. To use this cloud type, select the "Cleversafe-New" cloud type.

Note: Incorrect cloud type affects migration of large files that are transferred using multipart upload (MPU) technology.

--username UserName
Specifies the user name of the cloud object storage account.

Note: For Amazon S3 and IBM Cloud Object Storage, it represents the access key.

--pwd-file PasswordFile
Specifies a file that includes the password. For Amazon S3 and IBM Cloud Object Storage, password represents the secret access key.

--cloud-url CloudURL
Specifies the URL of the cloud storage provider.

Note: Optional for S3, but mandatory for Swift, Swift-Keystone, and IBM Cloud Object Storage. For object storage providers that require an end-point URL, the end-point URL need not have a container/vault name.

--tenant-id Tenant ID
Specifies the tenant ID for the cloud storage provider account.

Note: Optional for cloud type “S3” but mandatory for cloud types “Swift” and “Swift-Keystone”

--location Location
Specifies the preferred location of the object cloud storage provider. The valid values for S3 are the following:

- us-west-1
- us-west-2
- EU or eu-west-1
- eu-central-1

- apsoutheast-1
- ap-southeast-2
- ap-northeast-1
- ap-northeast-2
- sa-east-1

For IBM Cloud Object Storage, use the provisioning code for vault provisioning template. Do not use the name of the provisioning template. If a location is not used to provide provisioning code, then the default provisioning template is used.

--object size ObjectSize
Size of the objects to be transferred to the cloud for the pre-test.

create
Creates a cloud account definition to access the remote cloud.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--cloud-name CloudName
Specifies a name that uniquely identifies the cloud object storage account on the node.

Note: No special characters are allowed in the name except for "-" and "_".

--cloud-type CloudType
Specifies the name of the object storage provider. The following object storage providers are supported:

- Amazon Simple Storage Service (S3)
- OpenStack Swift (SWIFT)
- OpenStack Swift with Keystone authentication (Swift-Keystone)
- S3 API layer on Swift (Swift3)
- IBM Cloud Object Storage version earlier than 3.7.2
- IBM Cloud Object Storage version 3.7.2 and above

--username Name
Specifies the user name of the cloud object storage account.

Note: For Amazon S3 and IBM Cloud Object Storage, it represents the access key.

--pwd-file PasswordFile
Specifies a file that includes the password.

--enable {TRUE | FALSE}
Enables or disables the added cloud object storage account for Transparent Cloud Tiering node.

Note: This property is always enabled in the 4.2.1 release.

--cloud-url CloudURL
Specifies the URL of the cloud storage provider.

Note: Optional for S3, but mandatory for Swift, Swift-Keystone, and IBM Cloud Object Storage. For object storage providers that require an end-point URL, the end-point URL need not have a container/vault name.

--tenant-id Tenant ID
Specifies the tenant ID for the cloud storage provider account.

Note: Optional for cloud type "S3" but mandatory for cloud types "Swift" and "Swift-Keystone."

--location Location
Specifies the preferred location of the object cloud storage provider. The valid values for S3 are the following:

- us-west-1
- us-west-2
- EU or eu-west-1
- eu-central-1
- apsoutheast-1
- ap-southeast-2
- ap-northeast-1
- ap-northeast-2
- sa-east-1

For IBM Cloud Object Storage, use the provisioning code for vault provisioning template. Do not use the name of the provisioning template. If a location is not used to provide provisioning code, then the default provisioning template is used.

--meta-location MetaLocation
For IBM Cloud Object Storage, two containers are created on the cloud storage tier, one for data and one for metadata. You can use this attribute to specify different vault provisioning codes for metadata containers. Data container should have indexing disabled, whereas metadata container should have indexing enabled.

--mpu-parts-size MPUPartsSize
Specifies multi-part upload size in MB. Value that is allowed is 5 - 32, default being 32.

--server-cert-path ServerCertPath
Specifies the certificate path for the self-signed certificates that are presented by the private object storage servers. This is required only when the cloud URL uses https.

--enc-enable [TRUE | FALSE]
Specifies whether you want to enable encryption on the data that is transferred to the object storage.

--etag-enable [TRUE | FALSE]
Specifies whether you want to enable an integrity check on the data that is migrated or recalled to or from the cloud storage.

test
Verifies the validity of the cloud account.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--cloud-name CloudName
Specifies the unique name that was provided to the cloud object storage account on the Transparent Cloud Tiering node.

update
Updates the cloud account attributes (except the cloud type) that are provided while you create a cloud account. To update the cloud type, you must delete the cloud account and create a new one.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--cloud-name CloudName
Specifies the unique name that was provided to the cloud object storage account on the node.

--pwd-file PasswordFile
Specifies the file that includes the password.

--mpu-parts-size MPUPartsSize
Specifies multi-part upload size in MB. Value that is allowed is 5 - 32, default being 32.

--location Location
Specifies the preferred location that was provided while you created the object cloud storage tier.

--cloud-type CloudType
Specifies the name of the object storage provider.

delete
Deletes the cloud storage tier that is created by using the **mmcloudgateway account create** command.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--cloud-name CloudName
Specifies the cloud account that you want to delete.

list
Lists the registered cloud accounts. Displays more information about the configured cloud account such as the cloud provider name, cloud provider tenant ID, cloud provider URL.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--cloud-name CloudName
Specifies the cloud account that you want to delete.

config
Configures and tunes the Gateway node parameters with one of the following actions:

set
Sets the following system parameters, overriding the default values:

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--port Port
Specifies the port the Transparent Cloud Tiering service listens on incoming configuration and data migration/recall commands.

--slize-size SliceSize
Specifies the internal unit of transferring data within Transparent Cloud Tiering modules. Higher slice size indicates better performance. Default value is 256 KB.

--migrate-threadpool-size MigrateThreadpoolSize
Enables administrators to increase the parallelism for data migration within the Transparent Cloud Tiering service. Higher threadpool-size indicates better performance. Its value can be within 1 to 32, with default being 16.

--recall-threadpool-size RecallThreadpoolSize
Enables administrators to increase the parallelism for data migration within the Transparent Cloud Tiering service. Higher threadpool-size indicates better performance. Its value can be within 1 to 32, with default being 16.

--tracing-enable {TRUE | FALSE}
Tracing level is to set non-default tracing levels for various Transparent Cloud Tiering internal components to generate more debug data if any problems occur.

--audit-enable {TRUE | FALSE}
Enables administrator to change the default audit behavior to be able to record important events within the Transparent Cloud Tiering service.

--rotate-key
Enables administrator to generate a new key as and when needed according to the security requirements.

--rkm-enable
Enables the IBM Security Key Lifecycle Manager.

--rkm-servername RKMServerName
Specifies the host name or IP address of the IBM Security Lifecycle Manager server.

--rkm-port RKMPort
Specifies the port number on which the IBM Security Key Lifecycle Manager server listens for requests. Default value is 9080.

--rkm-username RKMUserName
Specifies the user name of the IBM Security Key Lifecycle Manager server REST Global Admin. Default value is SKLMAdmin.

--pwd-file PasswordFile
Specifies the password file of the IBM Security Key Lifecycle Manager server REST Global Admin.

unset
Unsets the configured attributes and retains the default values.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--slize-size SliceSize
Specifies the internal unit of transferring data within modules. Higher slice size indicates better performance. Default value is 256 KB.

--recall-threadpool-size
Enables administrators to increase the parallelism for data recall within the service. Higher threadpool-size indicates better performance. Its value can be within 1 to 32, with default being 16.

--tracing-enable
Tracing level is to set non-default tracing levels for various internal components to generate more debug data if any problems occur.

--tracing-level
Enables administrators to print trace messages of the internal components in a file.

--audit-enable
Enables administrator to change the default audit behavior to be able to record important events within the service.

list
Lists the current configurations such as IP address, port number, thread-pool size, tracing level, slice size.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

filesystem
Enables the mapping of a file system to a node class.

create
Specifies the node class and a container name.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--file-system FileSystem
Specifies the file system.

--container-prefix ContainerPrefix
Specifies the cloud container.

--override-container-name
Overrides the container name and make it unique by attaching the file system ID to the container prefix.

delete
Deletes the cloud file system association to the node class.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

--file-system FileSystem
Specifies the file system that is attached to the cloud storage tier.

list
Lists the table of file systems associated with the specified node class.

node
Enables administrators to manage registration of Transparent Cloud Tiering nodes within a cluster and also display the node class the nodes are part of.

list
Lists the Transparent Cloud Tiering nodes that are registered to the cluster.

--cloud-nodeclass CloudNodeClass
Specifies the node class that was created by using the **mmcrnodeclass** command.

service
Manages the Transparent Cloud Tiering service with these options:

start
Starts the Transparent Cloud Tiering service for a node or set of nodes and make the service available for file movement.

-N Specifies the nodes.

Node[,Node...]
Specifies the list of nodes where the service needs to be started.

NodeFile
Specifies a file, containing the list of nodes where the service needs to be started.

NodeClass
Specifies the node class.

stop
Stops the Transparent Cloud Tiering service for a node or set of nodes.

-N Specifies the nodes.

Node[,Node...]
Specifies the list of nodes where the service needs to be stopped.

NodeFile
Specifies a file, containing the list of nodes where the service needs to be started.

NodeClass
Specifies the node class.

status
Displays detailed status of the Transparent Cloud Tiering service including running state of the daemon service, cloud account name, and its connectivity status.

-N Specifies the nodes.

Node[,Node...]
Specifies the list of nodes where the status of the service needs to be checked.

NodeFile
Specifies a file, containing the list of nodes where the status of the service needs to be started.

NodeClass
Specifies the node class.

files
Manages the resident and co-resident files on the cloud tier, with the following options:

migrate
Migrates the specified files or file sets to the cloud storage tier.

-v Specifies the verbose message.

--File [File ...]
Specifies multiple files that need to be migrated to the cloud storage tier. This parameter must be a complete file name. It cannot be a fragment of a file name and it cannot be a path.

recall
Recalls the specified files or file sets from the cloud storage tier.

-v Specifies the verbose message.

--co-resident-state
Indicates that the files are co-resident on the cloud storage tier.

--File [File ...]
Specifies multiple files that need to be recalled from the storage tier. This parameter must be a complete file name. It cannot be a fragment of a file name and it cannot be a path.

restore
Restores a file or list of files from the cloud storage tier when the local files are lost.

-v Specifies the verbose message.

--overwrite
Overwrite the files if needed. If this option is not set, files will not be overwritten, and the files that are retrieved from the cloud will remain in temporary locations.

-F Loads file arguments from the given filename.

--dry-run
Queries the local database and prints what would have been sent to the server. Does not contact the server. This is intended for debugging.

--restore-location RestoreLocation
Specifies the target location of the files to be restored.

--id Id
Specifies the version ID of a file if the file has multiple versions.

File
Specifies the files to be restored.

delete
Deletes the specified files or file sets.

--delete-local-file
Deletes the local files and the corresponding cloud objects.

--recall-cloud-file
Recalls the files from the cloud before they are deleted on the cloud. The status of local files becomes resident after the operation.

--require-local-file
Removes the extended attributes from a co-resident file and makes it resident, without deleting the corresponding cloud objects. The option requires the file data to be present on the file system and will not work on a non-resident file.

--keep-last-cloud-file
This option deletes all the versions of the file except the last one from the cloud. For example, if a file has three versions on the cloud, then versions 1 and 2 are deleted and version 3 is retained.

--File [File ...]
Specifies multiple files. This parameter must be a complete file name. It cannot be a fragment of a file name and it cannot be a path.

reconcile
Reconciles files between your file system and the cloud storage tier.

Device
Specifies the device name associated with the file system.

all
Specifies all the device names.

cloudList
Lists the files on the cloud.

--path Path
Lists files and directories under the specified path.

--recursive
List all files in all directories under the current directory.

--depth Depth
List directories up to the specified depth under the specified path. Default is to list up to the full depth. Specify 0 to list only the current directory.

--file [File]
Specifies the names of the files that need to be listed. This parameter must be a complete file name. It cannot be a fragment of a file name and it cannot be a path.

--file-versions File
Displays information about all versions of the files specified by the full path.

--files-usage --path Path
Displays cloud data and metadata space usage under the specified path.

--reconcile-status --path Path
Displays the progress of the reconcile operation.

rebuildDB
Rebuilds the database manually.

Device
Specifies the device name associated with the file system whose database is corrupted and which is in need of manual recovery.

list
Lists the files and the associated states.

--File [File ...]
Specifies the names of the files that need to be listed. This parameter must be a complete file name. It cannot be a fragment of a file name and it cannot be a path.

Security

You must have root authority to run the **mmcloudgateway** 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To pre-validate the cloud storage settings, issue this command:

```
mmcloudgateway account pre-test --cloud-type SWIFT-KEYSTONE --username admin --pwd-file PFile
 --cloud-url http://10.11.12.13:5000/v2.0 --tenant-id admin
```

   The system displays output similar to this:

```
Validation under progress. This may take a while...
Cloud provider validation is under progress. Please wait for results to publish..
Cloud Provider Validator Tool Results:
-------------------------------------
Summary:
Provider compatibility with Transparent Cloud Tiering : Compatible
Ethernet Link Speed : 10000Mb/s
IOPS number for Puts : 14.0
IOPS number for Gets : 70.0
Data Throughput : 29 MB/s
-------------------------------------
Details:
Successful container operations : create, put, get
Successful object operations: put, get, getMeta, remove
Unsuccessful container operations : none
Unsuccessful object operations : none
-------------------------------------
mmcloudgateway: Command completed.
real 1m44.055s
user 0m40.488s
sys 0m1.999s
```

2. To view the registered nodes in a node class, issue this command:

```
mmcloudgateway node list
```

   The system displays output similar to this:

```
Node  Cloud node name                 Cloud Node Class
-----------------------------------------------------
    8  c35f1m4n09.gpfs.net             CloudNodesClass
    9  c34f2n06.gpfs.net               CloudNodesClass
```

3. To start the Transparent Cloud Tiering service on the node class *TCTNodeClass1*, issue this command:

```
mmcloudgateway service start -N TCTNodeClass1
```

   The system displays output similar to this:

```
Node  Daemon node name             TCT Server Status  TCT Filesystem Status  TCT Account Status
-------------------------------------------------------------------------------------------------
    9     c34f2n06.gpfs.net        Not Configured      Not Configured        Not Configured
    8     c35f1m4n09.gpfs.net      Not Configured      Not Configured        Not Configured
```

4. To verify the status of the Transparent Cloud Tiering service, issue this command:

```
mmcloudgateway service status -N TCTNodeClass1
```

   The system displays output similar to this:

```
Node  Daemon node name             TCT Server Status  TCT Filesystem Status  TCT Account Status
-------------------------------------------------------------------------------------------------
    9     c34f2n06.gpfs.net        Not Configured      Not Configured        Not Configured
    8     c35f1m4n09.gpfs.net      Not Configured      Not Configured        Not Configured
```

5. To create a file system association, issue this command:

```
mmcloudgateway filesystem create --cloud-nodeclass TCTNodeClass1 --filesystem /dev/gpfs0
 --container-prefix multinode
```

   The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on cf2u18.
mmcloudgateway: Command completed.
```

6. To create a cloud storage account with IBM Cloud Object Storage version 3.7.2 and above as cloud type, issue this command:

```
mmcloudgateway account create --cloud-nodeclass TCTNodeClass1 --cloud-name tctenew --cloud-type
 cleversafe-new --username "XYZ" --pwd-file PFile --enable TRUE --cloud-url http://10.1.9.41
```

   The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on c350f2u18.
mmcloudgateway: Command completed.
```

7. To create a cloud storage tier with Swift-Keystone as cloud type, issue this command:

```
mmcloudgateway account create --cloud-nodeclass TCTNodeClass1 --cloud-name  tct --cloud-type SWIFT-KEYSTONE
--username admin --pwd-file PFile --enable TRUE --cloud-url http://10.11.12.13:5000/v2.0
--tenant-id admin --enc-enable TRUE
--cloud-nodeclass cloud
```

   The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on c350f2u18.
mmcloudgateway: Command completed.
```

8. To create a cloud account for the S3 cloud type, issue a command similar to this:

```
mmcloudgateway account create --cloud-nodeclass TCTNodeClass1 --cloud-name cloudtest
 --cloud-type s3 --username admin --pwd-file MyFile --enable true
```

   The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on c350f2u18.
mmcloudgateway: Command completed.
```

9. To list the cloud accounts with cloud name, cloudtest, issue this command:

```
mmcloudgateway account list --cloud-nodeclass TCTNodeClass --cloud-name cloudtes
```

   The system displays output similar to this:

```
Configured Cloud Details:
Cloud Provider Name : cloudtest
Cloud Provider Tenant Id : null
Cloud Provider URL : http://9.114.98.4
Cloud Provider Type : cleversafe-new
Cloud Provider User Name : WNFpMgj93fkEYFbjoN16
Cloud Provider Enabled : true
Filesystem Root Path : /dev/gpfs0
Container : multinode2222
```

10. To update a cloud storage with an MPU part size of 32, issue this command:

    ```
    mmcloudgateway account update --cloud-nodeclass cloud --cloud-name mcstore --mpu-parts-size 32
    ```

    The system displays output similar to this:

    ```
    mmcloudgateway: Sending the Transparent Cloud Tiering request to the first sucessful server.
    mmcloudgateway: This may take a while...
    mmcloudgateway: Command completed successfully on c80f4m5n01.gpfs.net.
    mmcloudgateway: Command completed.
    ```

11. To verify that the cloud storage tier is active, issue this command:

    ```
    mmcloudgateway account test --cloud-nodeclass cloud --cloud-name mcstore
    ```

    The system displays output similar to this:

    ```
    mmcloudgateway: Sending the Transparent Cloud Tiering request to the first sucessful server.
    mmcloudgateway: This may take a while...
    Cloud Status : Configured cloud account is Active!
    mmcloudgateway: Command completed successfully on vmip206.gpfs.net.
    mmcloudgateway: Command completed.
    ```

12. To list the current cloud configuration, issue this command:

    ```
    mmcloudgateway config list
    ```

    The system displays output similar to this:

    ```
    audit.enable=true
    ip-address=9.47.83.202
    port=8085
    sliceSize=262144
    threadpool.poolSize=16
    tracing.enable=true
    tracing.level=ALL=4
    ```

13. To migrate a file (file1) to the configured cloud storage tier, issue this command:

    ```
    mmcloudgateway files migrate file1
    ```

    The system displays output similar to this:

    ```
    mmcloudgateway: Command completed.
    ```

14. To migrate multiple files (file1 and file2) to the configured cloud storage tier, issue this command:

    ```
    mmcloudgateway files migrate file1 file2
    ```

    The system displays output similar to this:

    ```
    mmcloudgateway: Command completed.
    ```

15. To verify that the file is migrated to the configured cloud storage tier, issue this command:

    ```
    mmcloudgateway files list file1
    ```

    The system displays output similar to this:

    ```
    File name    : /gpfs/girish/file1
    On-line size : 45
    Used blocks  : 0
    Data Version : 1
    Meta Version : 1
    State        : Non-resident
    Base Name    : 7448805A60ED1970.17F2AFD45704E1E4.52E20457CA532F09.0000000000000000.57B76BDD.000000000000100B
    ```

Note: The **State** is displaying as **Non-resident** . This means that the file is successfully migrated to the cloud storage tier.

16. To recall a file from the configured cloud storage tier, issue this command:

```
mmcloudgateway files recall file1
```

The system displays output similar to this:

```
mmcloudgateway: Command completed.
```

Note: If you run the **mmcloudgateway filesystem list** file1 command, the value of the **State** attribute is displayed as **Co-resident** . This means that the file is successfully recalled.

17. To recall multiple files (file1 and file2) from the configured cloud storage tier, issue this command:

```
mmcloudgateway files recall file1 file2
```

The system displays output similar to this:

```
mmcloudgateway: Command completed.
```

18. To delete the association between the file system and the cloud storage tier, issue this command:

```
mmcloudgateway filesystem delete --cloud-nodeclass TCTNodeClass1 --filesystem /dev/gpfs0
```

The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on c350f2u18.
mmcloudgateway: Command completed.
```

19. To delete a cloud storage account, issue this command:

```
mmcloudgateway account delete --cloud-nodeclass TCTNodeClass1 --cloud-name mycloud
```

The system displays output similar to this:

```
mmcloudgateway: Sending the Transparent Cloud Tiering request to the first successful server.
mmcloudgateway: This may take a while...
mmcloudgateway: Command completed successfully on c350f2u18.
mmcloudgateway: Command completed.
```

See also

- "mmchconfig command" on page 123
- "mmlscluster command" on page 357
- "mmchnode command" on page 180
- "mmlsconfig command" on page 359
- "mmnfs command" on page 402
- "mmobj command" on page 414
- "mmsmb command" on page 500
- "mmuserauth command" on page 527

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 on Windows nodes* 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:

```
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: 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* in *IBM Spectrum Scale: Administration Guide*.

-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**

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmlscluster
```

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 29
- "mmchconfig command" on page 123
- "mmdelnode command" on page 276
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359

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 **mmlinkfileset** command is issued. The **mmcrfileset** command is separate from the **mmlinkfileset** command to allow the administrator to establish policies and quotas on the fileset before it is linked into the namespace.

For information on filesets, see the *Filesets* section in the *IBM Spectrum Scale: 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*.

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.

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/thefs1 from node c41bn3, using protocol **nfs://**:

```
mmcrfileset fs3 singleWriter2  -p afmtarget=nfs://c41bn3/gpfs/thefs1/target2 -p afmmode=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/thefs1, using protocol **gpfs://**:

```
mmcrfileset fs3 singleWriter1  -p afmtarget=gpfs:///gpfs/thefs1/target1 -p afmmode=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: 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 disable.

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. 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 detailed information, see *AFM-based Asynchronous Disaster Recovery (AFM DR)* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For more information, see the topic about caching modes in the *IBM Spectrum Scale: 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.

no Specifies that the home snapshot link directory is not visible.

See *Peer snapshot -psnap* in *IBM Spectrum Scale: Concepts, Planning, and Installation 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).

setAclOnly
Specifies that permissions can be changed using ACL commands and API only (**chmod** will not be accepted).

chmodAndSetAcl
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.

chmodAndUpdateAcl
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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 163
- “mmdelfileset command” on page 271
- “mmlinkfileset command” on page 350
- “mmlsfileset command” on page 365
- “mmunlinkfileset command” on page 524

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:

1. Supported block sizes are 64 KiB, 128 KiB, 256 KiB, 512 KiB, 1 MiB, 2 MiB, 4 MiB, 8 MiB, and 16 MiB.
2. 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.
3. 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 IBM Spectrum Scale RAID is in use, the block size must equal the vdisk track size. For more information about IBM Spectrum Scale RAID, see *IBM Spectrum Scale RAID: Administration*.
4. 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/atsmastr.nsf/Web/WhitePapers).
5. 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:

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: 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 File Placement Optimizer (FPO) feature is to be enabled for the storage pool. For more information on FPO, see the *File Placement Optimizer* section in the *IBM Spectrum Scale: Administration Guide*.

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. The following conditions must be met while using a write affinity depth of 2 to get evenly allocated space in all disks:

1. The configuration in disk number, disk size, and node number for each rack must be similar.
2. The number of nodes must be the same in the bottom half and the top half of each rack.

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: 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.

-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 IBM Spectrum Protect hierarchical storage management (HSM) before the file system can be mounted. The default is **-z no**. For more information on DMAPI for GPFS, see “GPFS-specific DMAPI events” on page 565.

--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 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 169
- "mmdelfs command" on page 274
- "mmdf command" on page 287
- "mmedquota command" on page 298
- "mmfsck command" on page 304
- "mmlsfs command" on page 369
- "mmlspool command" on page 386

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 185
- "mmdelnodeclass command" on page 279
- "mmlsnodeclass command" on page 379

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 IBM Spectrum Scale RAID, a server list is not allowed. The servers are determined from the underlying vdisk definition. For more information about IBM Spectrum Scale RAID, see the *IBM Spectrum Scale 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.

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: 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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

To create two new NSDs from the stanza file `/tmp/newNSDstanza` containing:

```
%nsd: device=/dev/sdav1
  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
```

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 23
- "mmchnsd command" on page 188
- "mmcrfs command" on page 229
- "mmdeldisk command" on page 266
- "mmdelnsd command" on page 281
- "mmlsnsd command" on page 381
- "mmrpldisk command" on page 485

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 [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot[-j FilesetName[,FilesetName...]]
```

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.

In IBM Spectrum Scale Release 4.2.1 and later, snapshot commands support the specification of multiple snapshots. Users can easily create and delete multiple snapshots. Also, system performance is increased by batching operations and reducing overhead.

In this release, the following new usages of the **mmcrsnapshot** command have been introduced:

```
mmcrsnapshot device [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot...]
mmcrsnapshot device [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot...] -j Fileset
mmcrsnapshot device Snapshot -j Fileset1,Fileset2,...
```

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 information about protection against media failures, see the *Recoverability considerations* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For more information on global snapshots, see *Creating and maintaining snapshots of file systems* in the *IBM Spectrum Scale: Administration Guide*.

For more information on fileset snapshots, see *Fileset-level snapshots* in the *IBM Spectrum Scale: 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.

Fileset
The name of the fileset that contains the fileset snapshot to be created. If *Fileset* is not specified, the **mmcrsnapshot** command creates a global snapshot named Snapshot.

Note: Ensure that multiple snapshots and multiple filesets are not used together.

Snapshot
Specifies the name given to the snapshot.

The snapshot names are separated by a comma.

The snapshot specifier describes global and fileset snapshots. For example, *Fileset1*:*Snapshot1* specifies a fileset snapshot named Snapshot1 for fileset Fileset1. If *Fileset1* is empty, *Snapshot1* is a global snapshot named Snapshot1.

Each global snapshot name must be unique from any other global snapshots. If you do not want to traverse the root of the file system 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.

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, *Snapshot* 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.

Note:

- Ensure that the snapshot name does not include a colon (:) and a comma (,).
- Ensure that multiple snapshots and multiple filesets are not used together.

-j *FilesetName*
Creates a snapshot that includes the specified fileset and all the dependent filesets that share the same inode space. *FilesetName* refers to an independent fileset. If **-j** is not specified, the **mmcrsnapshot** command creates a global snapshot that includes all filesets.

If **-j** is specified, fileset names can be considered as a list separated by commas.

Note:

- When a list of snapshots separated by a comma (,) is used with the **-j** option, the fileset is applicable to each snapshot that does not use the colon (:) syntax. The fileset name must not consist of a white space.
- Only one global snapshot and one snapshot can be specified in each fileset. Therefore, multiple snapshots must not be created with the **-j** option. IBM recommends that at the most one global snapshot and one fileset snapshot must be included in each independent fileset. If multiple snapshots are specified with the **-j** option, a snapshot with the same name is created for each fileset that has been listed.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 device -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.
```

4. To create a snapshot for the fset1, fset2, fset3 filesets for the file system **fs1**, run the following command:

```
mmcrsnapshot fs1 snap1 -j fset1,fset2,fset3
```

The system displays the following output:

```
Flushing dirty data for snapshot fset1:snap1 fset2:snap1 fset3:snap1 (1..3) of 3...
Quiescing all file system operations.
Snapshot fset1:snap1 created with id 1.
Snapshot fset2:snap1 created with id 2.
Snapshot fset3:snap1 created with id 3.
```

5. To specify different snapshot names for each fileset (fset1, fset2, and fset3) for the file system **fs1**, run the following command:

```
mmcrsnapshot fs1 fset1:snapA,fset2:snapB,fset3:snapC
```

The system displays the following output:

```
Flushing dirty data for snapshot fset1:snapA fset2:snapB fset3:snapC (1..3) of 3...
Quiescing all file system operations.
Snapshot fset1:snapA created with id 4.
Snapshot fset2:snapB created with id 5.
Snapshot fset3:snapC created with id 6.
```

To view the list of snapshots, run the following command:

```
mmlssnapshot fs1
```

The system displays the following output:

```
Snapshots in file system fs1:
Directory               SnapId    Status  Created                   Fileset
snap1                   1         Valid   Thu May 12 04:27:18 2016  fset1
snap1                   2         Valid   Thu May 12 04:27:18 2016  fset2
snap1                   3         Valid   Thu May 12 04:27:18 2016  fset3
snapA                   4         Valid   Thu May 12 04:28:33 2016  fset1
snapB                   5         Valid   Thu May 12 04:28:33 2016  fset2
snapC                   6         Valid   Thu May 12 04:28:33 2016  fset3
```

See also

- "mmdelsnapshot command" on page 283
- "mmlssnapshot command" on page 395
- "mmrestorefs command" on page 471
- "mmsnapdir command" on page 511

Location

```
/usr/lpp/mmfs/bin
```

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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmlsquota -d -u gpfs1
```

The system displays information similar to:

```
         Default Block Limits(KB)          | Default File Limits
Filesystem type          quota      limit  |    quota    limit Remarks
gpfs1      USR        19922944   20971520  |     1024    20480
```

2. To set default quotas for new users of fileset **fset1** in file system **gpfs1**, issue this command:

```
mmdefedquota -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:

```
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     3145728    6291456  |        0        0  default on
gpfs1      fset1      GRP           0          0  |        0        0  default on
```

See also

- "mmchfs command" on page 169
- "mmcheckquota command" on page 159
- "mmcrfs command" on page 229
- "mmdefquotaoff command" on page 254
- "mmdefquotaon command" on page 257
- "mmedquota command" on page 298
- "mmlsfs command" on page 369
- "mmlsquota command" on page 391
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

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.

-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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

   ```
           Default Block Limits(KB)          | Default File Limits
   Filesystem type          quota      limit |    quota    limit Remarks
   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:

   ```
           Default Block Limits(KB)          | Default File Limits
   Filesystem type          quota      limit |    quota    limit Remarks
   fs0        GRP         no default limits

           Default Block Limits(KB)          | Default File Limits
   Filesystem type          quota      limit |    quota    limit Remarks
   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 159
- "mmdefedquota command" on page 251
- "mmdefquotaon command" on page 257
- "mmedquota command" on page 298
- "mmlsquota command" on page 391
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

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 **mmdefedquota** 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 **mmdefedquota** 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: 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 **mmdefquotaon** 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 Guide*.

GPFS must be running on the node from which the **mmdefquotaon** command is issued.

Examples

1. To activate default user quotas on file system **fs0**, issue this command:

```
mmdefquotaon -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:

```
mmdefquotaon -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
```

3. 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
```

4. 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:

```
            Default Block Limits(KB)   | Default File Limits
Filesystem Fileset 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 :

```
         Default Block Limits(KB)      | Default File Limits
Filesystem Fileset 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 159
- "mmchfs command" on page 169
- "mmcrfs command" on page 229
- "mmdefedquota command" on page 251
- "mmdefquotaoff command" on page 254
- "mmedquota command" on page 298
- "mmlsfs command" on page 369
- "mmlsquota command" on page 391
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

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 Guide*.

--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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To query the fragmentation state of file system **fs0**, issue this command:

   ```
   mmdefragfs fs0 -i
   ```

   The system displays information similar to:

   ```
   disk            disk size  in full  subblk in      %       %
   name          in nSubblk   blocks   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:

   ```
                       free subblk           free
     disk               in full            subblk in       %            %
     name               blocks    blk       fragments     free blk     blk util
                 before     after freed   before after 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
```

3. 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

                  free subblk              free
   disk            in full               subblk in          %             %
   name            blocks        blk     fragments        free blk     blk util
                 before    after 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 also

- "mmdf command" on page 287

Location

/usr/lpp/mmfs/bin

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 295
- "mmgetacl command" on page 317
- "mmputacl command" on page 450

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

To delete the **test1** callback from the GPFS system, issue this command:

```
mmmdelcallback 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 10
- "mmlscallback command" on page 355

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.

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 Guide*.

--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.

illCompressed
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.

illPlaced
Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

illReplicated
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
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.

--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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Example

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 sp1 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 'sp1'
tsdeldisk64 completed.
mmdeldisk: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

See also

- "mmadddisk command" on page 23
- "mmchdisk command" on page 151
- "mmlsdisk command" on page 361
- "mmrpldisk command" on page 485

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: 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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 163
- “mmcrfileset command” on page 223
- “mmlinkfileset command” on page 350
- “mmlsfileset command” on page 365
- “mmunlinkfileset command” on page 524

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 229
- "mmchfs command" on page 169
- "mmlsfs command" on page 369

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.
4. If the node is defined as a Transparent Cloud Tiering node. You can determine whether a node is a Transparent Cloud Tiering node by issuing the **mmcloudgateway node list** command. If the node is listed as the Transparent Cloud Tiering node, and you still want to delete it without deleting the cluster, first use the **mmchnode** command to disable the Transparent Cloud Tiering node role

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

You may issue the **mmdelnode** command from any node that will remain in the GPFS cluster.

Examples

1. 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.
   ```

2. 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 29
- "mmcrcluster command" on page 218
- "mmchconfig command" on page 123
- "mmlsfs command" on page 369
- "mmlscluster command" on page 357

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

To display the current user-defined node classes, issue this command:

```
mmlsnodeclass --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 239
- "mmchnodeclass command" on page 185
- "mmlsnodeclass command" on page 379

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 **mmlsnsd -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 **mmlsconfig tiebreakerDisks** command. Use the **mmchconfig** command to assign new tiebreaker disks prior to deleting NSDs from the cluster. For information on tiebreaker disks, see *Quorum* in *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 **mmlsnsd -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 Guide*.

-p *NSDId*
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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 COA8910B626630E
   ```

   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 241
- "mmlsnsd command" on page 381

Location

```
/usr/lpp/mmfs/bin
```

mmdelsnapshot command

Deletes a GPFS snapshot.

Synopsis

```
mmdelsnapshot Device [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot...
[-j FilesetName[,FilesetName...]][--qos QOSClass]
[-N{all | mount | Node[,Node...]|NodeFile | NodeClass
```

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: Administration Guide* for more information about file clones and policy files.

In IBM Spectrum Scale 4.2.1 and later, snapshot commands support the specification of multiple snapshots. Users can easily delete multiple snapshots for maintenance and cleanup operations. Also, system performance is increased by batching operations and reducing overhead.

In this release, the following new usages of the **mmdelsnapshot** command have been introduced:

```
mmdelsnapshot fs [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot...]
mmdelsnapshot fs [[Fileset]:]Snapshot[,[[Fileset]:]Snapshot...] -j Fileset
mmdelnapshot fs Snapshot -j Fileset1,Fileset2,...
```

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.

Fileset
: Specifies the name of the fileset that contains the fileset snapshot to be deleted. If *Fileset* is not specified, the **mmdelsnapshot** command deletes a global snapshot named *Snapshot*.

Note: Ensure that multiple snapshots and multiple filesets are not used together.

Snapshot
Specifies the name of the snapshot to be deleted.

The snapshot names are separated by a comma.

The snapshot specifier describes global and fileset snapshots. For example, *Fileset1*:*Snapshot1* specifies a fileset snapshot named Snapshot1 for fileset Fileset1. If *Fileset1* is empty, *Snapshot1* is a global snapshot named Snapshot1.

Note: Ensure that the snapshot name does not include a colon (:), a comma (,), and whitespaces. If the snapshot name consists of a whitespace, the whitespace must be quoted and part of the snapshot name. For example, if the snap1 and snap A snapshots must be deleted, use the command **'mmdelsnapshot pk2 "snap A, snap1"'**.

-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*.

Note: When a list of snapshots separated by a comma (,) is used with the **-j** option, the fileset is applicable to each snapshot that does not use the colon (:) syntax. The fileset name must not consist of a white space.

-N{all | mount | *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 Guide*.

--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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To delete the snapshot **snap1** for the file system **fs1**, run 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
```

2. To delete the snap1 snapshot from the fset1, fset2, and fset3 filesets for the file system **fs1**, run the following command:

```
mmdelsnapshot fs1 snap1 -j fset1,fset2,fset3
```

   The system displays the following output:

```
Invalidating snapshot files in fset1:snap1 fset2:snap1 fset3:snap1 (1..3) of 3...
Deleting files in snapshot fset1:snap1...
 100.00 % complete on Thu May 12 04:43:07 2016  (    100032 inodes with total          0 MB data processed)
Invalidating snapshot files in fset1:snap1/F/...
Deleting files in snapshot fset2:snap1...
 100.00 % complete on Thu May 12 04:43:07 2016  (    100032 inodes with total          0 MB data processed)
Invalidating snapshot files in fset2:snap1/F/...
Deleting files in snapshot fset3:snap1...
 100.00 % complete on Thu May 12 04:43:07 2016  (    100032 inodes with total          0 MB data processed)
Invalidating snapshot files in fset3:snap1/F/...
Delete snapshot fset1:snap1 successful.
Delete snapshot fset2:snap1 successful.
Delete snapshot fset3:snap1 successful.
```

3. To specify the snapshot names that must be deleted from each fileset in the file system **fs1**, run the following command:

```
mmdelsnapshot fs1 fset1:snapA,fset2:snapB,fset3:snapC
```

   The system displays the following command:

```
Invalidating snapshot files in fset3:snapC fset1:snapA fset2:snapB (1..3) of 3...
Deleting files in snapshot fset1:snapA...
 100.00 % complete on Thu May 12 04:44:46 2016  (    100032 inodes with total          0 MB data processed)
Invalidating snapshot files in fset1:snapA/F/...
```

```
|     Deleting files in snapshot fset2:snapB...
|      100.00 % complete on Thu May 12 04:44:46 2016  (    100032 inodes with total          0 MB data processed)
|     Invalidating snapshot files in fset2:snapB/F/...
|     Deleting files in snapshot fset3:snapC...
|      100.00 % complete on Thu May 12 04:44:46 2016  (    100032 inodes with total          0 MB data processed)
|     Invalidating snapshot files in fset3:snapC/F/...
|     Delete snapshot fset1:snapA successful.
|     Delete snapshot fset2:snapB successful.
|     Delete snapshot fset3:snapC successful.
```

See also

- "mmclone command" on page 201
- "mmcrsnapshot command" on page 246
- "mmlssnapshot command" on page 395
- "mmrestorefs command" on page 471
- "mmsnapdir command" on page 511

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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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

1. 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        disk size  failure holds    holds  free KB in  free KB in
   name            in KB    group metadata data  full blocks  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%)
   ```

2. 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
   ```

The system displays information similar to:

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

3. 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 169
- "mmcrfs command" on page 229
- "mmdelfs command" on page 274
- "mmlsfs command" on page 369

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

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.

--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 *Trace facility* in *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
0x2AAAAB103990 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

mmeditacl command

Creates or changes a GPFS access control list.

Synopsis

mmeditacl [**-d**] [**-k** {**nfs4** | **posix** | **native**}] *Filename*

Availability

Available on all IBM Spectrum Scale editions.

Description

Use the **mmeditacl** 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: 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, **mmeditacl** 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 **mmeditacl -k nfs4** always produces an NFS V4 ACL.
3. The command **mmeditacl -k posix** always produces a traditional ACL.
4. The command **mmeditacl -k native** always shows the ACL in its true form regardless of the file system setting.

The following describes how **mmeditacl** works for POSIX and NFS V4 ACLs:

```
Command              ACL    mmcrfs -k  Display        -d (default)
-------------------  -----  ---------  -------------  --------------
mmeditacl            posix  posix      Access ACL     Default ACL
mmeditacl            posix  nfs4       NFS V4 ACL     Error[1]
mmeditacl            posix  all        Access ACL     Default ACL
mmeditacl            nfs4   posix      Access ACL[2]  Default ACL[2]
mmeditacl            nfs4   nfs4       NFS V4 ACL     Error[1]
mmeditacl            nfs4   all        NFS V4 ACL     Error[1]
mmeditacl -k native  posix  any        Access ACL     Default ACL
mmeditacl -k native  nfs4   any        NFS V4 ACL     Error[1]
mmeditacl -k posix   posix  any        Access ACL     Default ACL
mmeditacl -k posix   nfs4   any        Access ACL[2]  Default ACL[2]
mmeditacl -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 **mmlsfs** 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).

Examples

To edit the ACL for a file named **project2.history**, issue this command:

```
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.

See also

- "mmdelacl command" on page 263
- "mmgetacl command" on page 317
- "mmputacl command" on page 450

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 Guide*.

The EDITOR environment variable must contain a complete path name, for example:

```
export EDITOR=/bin/vi
```

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
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 159
- "mmdefedquota command" on page 251
- "mmdefquotaoff command" on page 254
- "mmdefquotaon command" on page 257
- "mmlsquota command" on page 391
- "mmquotaon command" on page 455
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

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 IBM Spectrum Scale RAID-based file system, then all affected IBM Spectrum Scale 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 IBM Spectrum Scale 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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 336

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: 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.
- 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 **Fileset***FilesetId* 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: 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: 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" on page 196. 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: 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.

64 Do a full offline file system check to verify the integrity of the file system.

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 Guide*.

Examples

1. The following command checks file system `fs1` but does not try to repair inconsistencies:

   ```
   mmfsck fs1 -n
   ```

   The command displays output similar to the following. It completes without finding any inconsistencies in the file system:

   ```
   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.
```

2. The following command checks file system fs2 and repairs inconsistencies:

```
mmfsck fs2 -y
```

The command displays output similar to the following:

```
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.

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. The following command checks file system FSchk and records file system inconsistencies in the patch file path-towrite-patchfile:

   ```
   mmfsck FSchk -nv --patch-file path-towrite-patchfile
   ```

   The command displays output similar to the following:

   ```
   Creating patch file "path-towrite-patchfile" on node "Node3"

Checking "FSchk"
     fsckFlags                     0x8000009
     Stripe group manager          <c0n3>
     needNewLogs                   0


Error in inode 3670016 snap 0: has nFullBlocks field as 6
 Correct to 4? No
Checking for the first reference to a duplicate fragment
Error in inode 3670016 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376144 is a duplicate address
 Delete disk address? No
Cannot fix lost blocks if not deleting duplicate address.

Error in inode 3670016 snap 0: has nFullBlocks field as 6
 Correct to 4? No

Error in inode 3670016 snap 0: has lastBlockSubblocks field as 1
 Correct to 32? No
Node 192.168.200.69 (js23n19) ending inode scan 655872 to 1311743

Error in inode 3149568 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376128 is a duplicate address
 Delete disk address? No

Error in inode 3149568 snap 0: has lastBlockSubblocks field as 5
 Correct to 32? No
```

```
Error in inode 4720512 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376176 is a duplicate address
 Delete disk address? No

Error in inode 4720512 snap 0: has lastBlockSubblocks field as 1
 Correct to 32? No

Error in inode 5242880 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376192 is a duplicate address
 Delete disk address? No
...
Error in inode 4196224 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376160 is a duplicate address
 Delete disk address? No

Corrections are needed in the inode allocation map.
 Correct the allocation map? No
 Inode 5242881 is not in use but marked as used (0) in map.
1 inodes are not in use but marked.
Checking ACL file records

Error in inode 4718595 snap 0: Directory block 0 has entry with incorrect generation number
 file entry inode 4720512 "file.0"
 Delete entry? No

Error in inode 5242883 snap 0: Directory block 0 has entry referring to a deleted inode
 subdir entry inode 5242881 "direct"
 Delete entry? No

Error in inode 5242883 snap 0: has nlink field as 3
 Correct to 2? No

Checking file system replication status

No.   SnapId     InodeNum      FileType            Fix Error(s)
----- ---------- ------------- ------------------- --- ----------------------
1     0          3670016       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
2     0          3149568       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
3     0          4720512       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
                                                       + OrphanInode
4     0          5242880       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
5     0          4196224       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
6     0          4718595       Root Directory      N   DirectoryEntry
7     0          5242883       Root Directory      N   InodeMetadata
                                                       + DirectoryEntry
8     0          5242882       User                N   OrphanInode

           5246976 inodes
                   74   allocated
                    7   repairable
                    0   repaired
                    0   damaged
                    0   deallocated
                    2   orphaned
                    0   attached
                    0   corrupt ACL references

             89391104 subblocks
               326492   allocated
                    4   unreferenced
                    5   duplicates
```

```
                     0   deletable
                     0   deallocated

                  9726 addresses
                     0   suspended
                     5   duplicates
                     0   reserved file holes found
                     0   reserved file holes repaired
File system contains unrepaired damage.
Exit status 0:0:8.
```

4. The following command repairs the inconsistencies in the file system that are recorded in patch file `path-towrite-patchfile`. The patch file was created in Example 3:

```
mmfsck FSchk -v --patch-file path-towrite-patchfile --patch
```

The command displays information similar to the following:

```
   Checking "FSchk"
   fsckFlags                    0x1D00000A
   Stripe group manager         <c0n3>
   needNewLogs                  0

Checking patch file
Scanning patch file for pre-mount patches

Error in inode 3670016 snap 0: has nFullBlocks field as 6
 Correct to 4? Yes

Error in inode 3670016 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376144 is a duplicate address
 Delete disk address? Yes

Error in inode 3670016 snap 0: has nFullBlocks field as 6
 Correct to 4? Yes
 nFullBlocks existing value 4 does not match expected value 6.
 Skipping patch.

Error in inode 3670016 snap 0: has lastBlockSubblocks field as 1
 Correct to 32? Yes

Error in inode 3149568 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376128 is a duplicate address
 Delete disk address? Yes

Error in inode 3149568 snap 0: has lastBlockSubblocks field as 5
 Correct to 32? Yes

Error in inode 4720512 snap 0: inode has bad disk addr at offset 8
 replica 0 addr  2:3376176 is a duplicate address
 Delete disk address? Yes

Error in inode 4720512 snap 0: is unreferenced
 Attach inode to lost+found of fileset fset_4 filesetId 9? Yes

Error in inode 5242882 snap 0: is unreferenced
 Attach inode to lost+found of fileset fset_5 filesetId 10? Yes
 100 % complete on Thu May  5 04:20:58 2016

No.   SnapId     InodeNum      FileType            Fix Error(s)
----- ---------- ------------- ------------------- --- -----------------------
1     0          3670016       User                N   InodeMetadata
                                                       + DuplicateDiskAddr
2     0          3149568       User                Y   InodeMetadata
                                                       + DuplicateDiskAddr
3     0          4720512       User                Y   InodeMetadata
                                                       + DuplicateDiskAddr
                                                       + OrphanInode
```

```
|      4     0          5242880       User               Y   InodeMetadata
|                                                             + DuplicateDiskAddr
|      5     0          4196224       User               Y   InodeMetadata
|                                                             + DuplicateDiskAddr
|      6     0          4718595       Root Directory     Y   DirectoryEntry
|      7     0          5242883       Root Directory     Y   InodeMetadata
|                                                             + DirectoryEntry
|      8     0          5242882       User               Y   OrphanInode
|      File system is patched.
```

See also

- "mmcheckquota command" on page 159
- "mmcrfs command" on page 229
- "mmdelfs command" on page 274
- "mmdf command" on page 287
- "mmlsfs command" on page 369

Location

/usr/lpp/mmfs/bin

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: 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: 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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 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: 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::rwxc
   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
   ```

See also

- "mmeditacl command" on page 295
- "mmdelacl command" on page 263
- "mmputacl command" on page 450

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 Guide*.

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

more information, see *Requirements for administering a GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 123
- "mmcrcluster command" on page 218

- "mmshutdown command" on page 498
- "mmstartup command" on page 515

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
```

mmhealth command

Monitors health status of nodes.

Synopsis

```
mmhealth node show [ GPFS | NETWORK [ UserDefinedSubComponent ]
                   | FILESYSTEM [UserDefinedSubComponent ] | DISK | CES | AUTH | AUTH_OBJ
                   | BLOCK | CESNETWORK | NFS | OBJECT | SMB | CLOUDGATEWAY | GUI
                   | PERFMON ] [-N {Node[,Node..] | NodeFile | NodeClass}]
                  [--verbose] [--unhealthy]
```

or

```
mmhealth node eventlog [[--hour | --day | --week | --month] | [--verbose]]
```

Availability

Available with IBM Spectrum Scale Express Edition or higher.

Description

Use the **mmhealth** command to monitor the health of the node and services hosted on the node in IBM Spectrum Scale.

By using this command, IBM Spectrum Scale the administrator can monitor the health of each node and services hosted on that node. This command also shows the events that are responsible for the unhealthy status of the services hosted on that node. This data might be helpful for monitoring and analyzing the reasons for the unhealthy status of the node. So, **mmhealth** command acts as a problem determination tool to identify which services of the node are unhealthy and events responsible for their unhealthy status.

For more information about the system monitoring feature, see *IBM Spectrum Scale: Administration Guide*

Parameters

node
Displays the health status, specifically, at node level.

show
Displays the health status of the specified component with:

GPFS | NETWORK | FILESYSTEM | DISK | CES | AUTH | AUTH_OBJ | BLOCK | CESNETWORK | NFS | OBJECT | SMB | CLOUDGATEWAY | GUI | PERFMON
Displays the detailed health status of the specified component.

UserDefinedSubComponent
Displays services that are named by the customer, categorized by one of the other hosted services. For example, a file system named gpfs0 is a subcomponent of file system.

-N Allows the system to make remote calls to the other nodes in the cluster for:

Node[,Node....]
Specifies the node or list of nodes that must be monitored for the health status.

NodeFile
Specifies a file, containing a list of node descriptors, one per line, to be monitored for health status.

NodeClass
Specifies a node class that must be monitored for the health status.

--verbose
Shows the detailed health status of a node, including its sub-components.

--unhealthy
Displays the unhealthy components only.

eventlog
Shows the event history for a specified period of time. If no time period is specified, it displays all the events by default:

[--hour | --day | --week| --month]
Displays the event history for the specified time period.

[--verbose]
Displays additional information about the event like component name and event ID in the eventlog.

Exit status

0 Successful completion.

nonzero
A failure has occurred.

Security

You must have root authority to run the **mmhealth** 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 information about the requirements for administering a GPFS system in the *IBM Spectrum Scale: Administration Guide*.

Examples

1. To show the health status of the current node:

```
mmhealth node show
```

   The system displays output similar to this:

```
Node name:      test_node
Node status:    HEALTHY
Status Change:  39 min. ago

Component          Status        Reasons
----------------------------------------
GPFS               HEALTHY       -
NETWORK            HEALTHY       -
FILESYSTEM         HEALTHY       -
DISK               HEALTHY       -
CES                HEALTHY       -
PERFMON            HEALTHY       -
```

2. To view the health status of a specific node, issue this command:

```
mmhealth node show -N test_node2
```

   The system displays output similar to this:

```
Node name:      test_node2
Node status:    CHECKING
Status Change:  Now

Component       Status        Status Change    Reasons
-----------------------------------------------------------------
GPFS            CHECKING      Now              -
NETWORK         HEALTHY       Now              -
FILESYSTEM      CHECKING      Now              -
DISK            CHECKING      Now              -
CES             CHECKING      Now              -
PERFMON         HEALTHY       Now              -
```

3. To view the health status of all the nodes, issue this command:

```
mmhealth node show -N all
```

   The system displays output similar to this:

```
Node name:    test_node
Node status:  DEGRADED

Component           Status        Status Change     Reasons
-------------------------------------------------------------
GPFS                HEALTHY          Now             -
CES                 FAILED           Now             smbd_down
FileSystem          HEALTHY          Now             -

Node name:            test_node2
Node status:          HEALTHY

Component           Status        Status Change    Reasons
-----------------------------------------------------------
GPFS                HEALTHY       Now              -
CES                 HEALTHY       Now              -
FileSystem          HEALTHY       Now              -
```

4. To view the detailed health status of the component and its sub-component, issue this command:

```
mmhealth node show ces
```

   The system displays output similar to this:

```
Node name:    test_node

Component           Status        Reasons
-----------------------------------------------
CES                 FAILED        smbd_down
    AUTH               HEALTHY       -
    AUTH_OBJ           HEALTHY       -
    NFS                HEALTHY       -
    OBJ                HEALTHY       -
    SMB                FAILED        smbd_down

Event              Parameter       Severity       Description
-----------------------------------------------------------------
smbd_down          SMB             ERROR          SMBD process not running
```

5. To view the health status of only unhealthy components, issue this command:

```
mmhealth node show --unhealthy
```

The system displays output similar to this:

```
Node name:    test_node
Node status:  DEGRADED

Component           Status        Reasons
-----------------------------------------------
CES                 FAILED        smbd_down
```

6. To view the health status of sub-components of a node's component, issue this command:

```
mmhealth node show --verbose
```

The system displays output similar to this:

```
Node name:    test_node
Node status:  DEGRADED

Component           Status        Reasons
-----------------------------------------------
GPFS                HEALTHY       -
CES                 FAILED        smbd_down
    AUTH               HEALTHY       -
    AUTH_OBJ           HEALTHY       -
    NFS                HEALTHY       -
    OBJ                HEALTHY       -
    SMB                FAILED     smbd_down

FILESYSTEM           HEALTHY      -
    gpfs0              HEALTHY      -
    FSII               HEALTHY      -
```

7. To view the eventlog history of the node for the last hour, issue this command:

```
mmhealth node eventlog --hour
```

The system displays output similar to this:

```
Timestamp                       Event Name              Severity   Details
2016-04-07 10:12:24.394569 CEST       quorum_warn             WARNING    GPFS quorum monitoring returned unknown result
2016-04-07 10:12:39.366279 CEST       quorum_warn             WARNING    GPFS quorum monitoring returned unknown result
2016-04-07 10:12:54.356577 CEST       quorum_warn             WARNING    GPFS quorum monitoring returned unknown result
```

8. To view the eventlog history of the node for the last hour, issue this command:

```
mmhealth node eventlog --hour --verbose
```

The system displays output similar to this:

```
Timestamp                       Component     Event Name              Event ID Severity   Details
2016-04-07 10:12:54.356577 CEST        gpfs          quorum_warn              999291   WARNING   GPFS quorum monitoring returned unknown result
2016-04-07 10:13:09.359602 CEST        gpfs          quorum_warn              999291   WARNING   GPFS quorum monitoring returned unknown result
2016-04-07 10:13:24.425680 CEST        gpfs          quorum_warn              999291   WARNING   GPFS quorum monitoring returned unknown result
```

See also

- "mmchconfig command" on page 123
- "mmlscluster command" on page 357
- "mmobj command" on page 414
- "mmsmb command" on page 500

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]
            [--qos QOSClass] [--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 Guide*.

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 **EXCLUDE**d file and the applicable rule.

5 Displays the same information as 4, plus the attributes of candidate and **EXCLUDE**d 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 Guide*.

-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 IBM Spectrum Protect through the **dsmc** commands.

--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" on page 196. 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: Administration Guide*.

--tsm-server *ServerName*
Specifies the server name to provide to the IBM Spectrum Protect **dsmc** command used to store the image data files in the IBM Spectrum Protect 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.

-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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 IBM Spectrum Protect server, issue:

```
mmimgbackup fs9 -S snap1 -g /backup_images --image sobar.cluster.fs9
```

To show that the images are stored on the IBM Spectrum Protect 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 56
- "mmbackupconfig command" on page 81
- "mmimgrestore command" on page 333
- "mmrestoreconfig command" on page 467

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] [--qos QOSClass] [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)* in *IBM Spectrum Scale: Administration Guide*.

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 **EXCLUDE**d file and the applicable rule.

5 Displays the same information as 4, plus the attributes of candidate and **EXCLUDE**d 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.

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 Guide*.

--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.

--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" on page 196. 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: Administration Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 56
- "mmbackupconfig command" on page 81
- "mmimgbackup command" on page 329
- "mmrestoreconfig command" on page 467

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 **mmlsnsd** 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 **mmlsdisk** 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.

If the specified file system device is a IBM Spectrum Scale RAID-based file system, then all affected IBM Spectrum Scale 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 IBM Spectrum Scale 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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: 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 302

Location

```
/usr/lpp/mmfs/bin
```

mmkeyserv command

Manages encryption key servers and clients.

Synopsis

```
mmkeyserv server add
   ServerName [--port RestPortNumber] [--user-id RestUserID]
   [--server-pwd PasswordFile] [--accept]
   [--backup ServerName[,ServerName...] [--distribute | --nodistribute]
   [--timeout ConnectionTimeout] [--retry ConnectionAttempts]
   [--interval Microseconds]
```

or

```
mmkeyserv server delete ServerName
```

or

```
mmkeyserv server show [ServerName]
```

or

```
mmkeyserv tenant add TenantName
    --server ServerName [--server-pwd PasswordFile]
```

or

```
mmkeyserv tenant delete TenantName
   --server ServerName [--server-pwd PasswordFile]
```

or

```
mmkeyserv tenant show [TenantName] [--server ServerName]
```

or

```
mmkeyserv key create --server ServerName [--server-pwd PasswordFile]
   --tenant TenantName [--count NumberOfKeys]
```

or

```
mmkeyserv key delete --server ServerName [--server-pwd PasswordFile]
   {--all --tenant TenantName | --file ListOfKeysFile}
```

or

```
mmkeyserv key show --server ServerName [--server-pwd PasswordFile]
   --tenant TenantName
```

or

```
mmkeyserv client create ClientName
   --server ServerName [--server-pwd PasswordFile]
   [--keystore-pwd PasswordFile]
```

or

```
mmkeyserv client delete ClientName
```

or

```
mmkeyserv client register ClientName
   --rkm-id RkmID --tenant TenantName [--server-pwd PasswordFile]
```

or

```
mmkeyserv client deregister ClientName
   --tenant TenantName [--server-pwd PasswordFile]
```

or

mmkeyserv client show [*ClientName* | **--server** *ServerName*]

or

mmkeyserv rkm change RkmID {[**--rkm-id** *NewRkmID*]
[**--backup** *ServerName*[,*ServerName...*]] [**--distribute** | **--nodistribute**]
[**--timeout** *ConnectionTimeout*] [**--retry** *ConnectionAttempts*]
[**--interval** *Microseconds*]}

or

mmkeyserv rkm show

Availability

Available with IBM Spectrum Scale Advanced Edition.

Description

With the **mmkeyserv** command, you can configure a cluster and a remote key manager (RKM) server so that nodes in the cluster can retrieve master encryption keys when they need to. You must set up an RKM server before you run this command. The RKM server software must be IBM Security Key Lifecycle Manager (ISKLM). Nodes in the cluster must have direct network access to the RKM server.

With this command you can connect to an RKM server, create GPFS tenants, create encryption keys, and create and register key clients. The command automatically generates and exchanges certificates and sets up a local keystore. You can also use this command to securely delete key clients, encryption keys, and tenants. You can run this command from any node in the cluster. Each node has a configuration file and a copy of the local keystore. Configuration changes affect all nodes in the cluster.

Password files: Several of the command options require a password file as a parameter. A password file is a text file that contains a password at the beginning. A password must be 1-20 characters in length. Because the password file is a security-sensitive file, it must have the following characteristics:

- It must be a regular file.
- It must be owned by the root user.
- Only the root user must have permission to read or write it.

The following terms are used:

IBM Security Key Lifecycle Manager (ISKLM)
: Required key management server software.

Master encryption key (MEK)
: A key for encrypting file encryption keys.

Remote key management (RKM) server
: A server with software that authenticates clients and provides them with master encryption keys.

RKM.conf file
: A configuration file that the mmkeyserv command maintains. Each node in the cluster has a copy of this file. The full path is `/var/mmfs/ssl/keyServ/RKM.conf`.

RKM stanza
: A block of configuration information that describes a registered key client. RKM stanzas are stored in the `RKM.conf` file.

RKM ID
: An identifier for an RKM stanza.

tenant A device group in ISKLM that contains MEKs for registered key clients.

Parameters

server
Manages a connection with an RKM server.

add
Adds an RKM server connection to the IBM Spectrum Scale cluster. You can adjust the values of some of these parameters later with the **mmkeyserv rkm change** command.

ServerName
Specifies the host name or IP address of the RKM server.

--port *RestPortNumber*
Specifies the port number for the Representational State Transfer (REST) interface. The default value is 9080.

--user-id *RestUserID*
Specifies the user ID for the RKM server. The default value is `SKLMAdmin`.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

--accept
Configures the command to automatically accept certificates from the RKM server. The acceptance prompt is suppressed.

--backup *ServerName*[**,***ServerName...*]
Specifies a comma-separated list of server names that you want to add to the list of backup RKM servers in the `RKM.conf` file. If an IBM Spectrum Scale node cannot retrieve a master encryption key from its main RKM server, it tries each backup server in the list until it either retrieves a key or exhausts the list.

Note: The **mmkeyserv** command itself does not attempt to contact backup servers or to replicate client information across background servers. The system administrator is responsible for maintaining replication across backup servers.

--distribute | **--nodistribute**

--distribute
Attempts to arrange the list of RKM server names (main RKM server and backup RKM servers) in the `RKM.conf` file in a different order on each node, so that each node connects with the servers in a different order. This option provides some performance advantage in retrieving MEKs. This option is the default.

--nodistribute
Does not attempt to arrange the list of backup RKM server names in the `RKM.conf` file.

--timeout *ConnectionTimeout*
Sets the connection timeout in seconds for retrieving an MEK from an RKM server. The valid range is 1 - 120 seconds.

--retry *ConnectionAttempts*
Sets the number of attempts to retry a connection to an RKM server. The valid range is 1 - 10 retries.

--interval *Microseconds*
Specifies the number of microseconds to wait between connection retries. The valid range is 1 - 1000000000.

delete
Removes a connection with an RKM server from the cluster.

ServerName
Specifies the host name or IP address of the RKM server that you want to disconnect from.

show
Displays information about RKM servers. The following table shows the display options:

Table 12. mmkeyserv server show

Option	Displays information about
show	All servers.
show *ServerName*	The specified server.

ServerName
Specifies the host name or IP address of an RKM server.

tenant
Manages tenants on RKM servers. A tenant is an ISKLM device group for holding encryption keys.

add
Specifies the name of a tenant to add to the IBM Spectrum Scale cluster.

- If the tenant is already added to the cluster, the command returns with an error.
- If the tenant exists on the RKM server but is not added to the cluster, the command adds the tenant to the cluster.
- If the tenant does not exist on the RKM server, the command creates the tenant on the server and adds the tenant to the cluster.

TenantName
Specifies the name of the tenant that you want to create.

--server *ServerName*
Specifies the name of the RKM server to which the tenant belongs.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

delete
Deletes a tenant from an RKM server.

Note:

- If you delete a tenant that has encryption keys on the key server, the command deletes the tenant from the cluster configuration but not from the key server.
- If you delete a tenant that has no encryption keys on the key server, the command deletes the tenant from both the cluster configuration and the key server.

TenantName
Specifies the name of the tenant that you want to delete.

--server *ServerName*
Specifies the name of the RKM server to which the tenant belongs.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

show
Displays information about tenants and RKM servers. The following table shows the results of various combinations of options:

Table 13. mmkeyserv tenant show

Option	Displays information about
show	All tenants from all RKM servers.
show *TenantName*	The specified tenant.
show ---server *ServerName*	All tenants from the specified RKM server.
show *TenantName* **--server** *ServerName*	The specified tenant and the specified server.

TenantName
Specifies the name of a tenant.

--server *ServerName*
Specifies the name of an RKM server.

key
Manages encryption keys.

create
Creates encryption keys in a tenant and displays the key IDs on the console.

Note: Make a note of the key IDs. You must specify an encryption key ID and an RKM ID when you write an encryption policy rule.

--server *ServerName*
Specifies the host name or IP address of an RKM server.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

--tenant *TenantName*
Specifies the name of the tenant in which you want to create the encryption keys.

--count *NumberOfKeys*
Specifies the number of keys to create. The default value is 1.

delete
Deletes encryption keys from a tenant.

CAUTION:
When you delete an encryption key, any data that was encrypted by that key becomes unrecoverable.

--server *ServerName*
Specifies the host name or IP address of an RKM server.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

--all --tenant *TenantName*
Deletes all the encryption keys in the specified tenant.

--file *ListOfKeysFile*
Specifies a file that contains a list of the key IDs of encryption keys that you want to delete, one key per line.

show
Displays information about the encryption keys in a tenant.

--server *ServerName*
Specifies the host name or IP address of an RKM server.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

--tenant *TenantName*
Specifies the name of the tenant that contains the keys that you want to display.

client
Manages key clients. The following facts are important:

- You need only one key client per cluster per RKM server. However, you can create and use multiple key clients on the same RKM server.
- You can register only one key client per tenant per cluster. However, you can register one key client to more than one tenant in the same RKM server.

create
Creates a key client to communicate with the RKM server.

ClientName
Specifies the name of the key client that you want to create. A key client name must be 1-16 characters in length and must be unique within an IBM Spectrum Scale cluster.

--server *ServerName*
Specifies the name of the RKM server to which the key client belongs.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

--keystore-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

delete
Deletes a key client.

ClientName
Specifies the name of the key client that you want to delete.

register
Registers a key client to a tenant.

ClientName
Specifies the name of the key client that you want to register.

--rkm-id *RkmID*
Specifies a new RKM ID. An RKM ID must be unique within the cluster, must be 1-21 characters in length, and can contain only alphanumeric characters or underscore (_). It must begin with a letter or an underscore. An RKM ID identifies an RKM stanza in the `RKM.conf` file. The stanza contains the information that a node needs to retrieve a master encryption key (MEK) from an RKM.

--tenant *TenantName*
Specifies the name of the tenant to which you want to register the client.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the

requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

deregister
Unregisters a key client from a tenant.

ClientName
Specifies the name of the key client that you want to unregister.

--tenant *TenantName*
Specifies the name of the tenant that you want to unregister the key client from.

--server-pwd *PasswordFile*
Specifies a password file that contains a password for accessing the RKM server. See the requirements for password files in the Description section of this topic. If this parameter is omitted, the command prompts for a password.

show
Displays information about a key client.

ClientName
Specifies the name of the client whose information you want to display.

--server *ServerName*
Specifies the name of the server to which the client belongs.

rkm

change
Changes the properties of an RKM stanza. For more information about an RKM stanza, see the Description section of this topic.

RkmID
Specifies the RKM ID of the stanza whose properties you want to change.

--rkm-id *RkmID*
Specifies a new RKM ID. An RKM ID must be unique within the cluster, must be 1-21 characters in length, and can contain only alphanumeric characters or underscore (_). It must begin with a letter or an underscore. An RKM ID identifies an RKM stanza in the `RKM.conf` file. The stanza contains the information that a node needs to retrieve a master encryption key (MEK) from an RKM.

--backup *ServerName*[,*ServerName...*]
Specifies a comma-separated list of server names that you want to add to the list of backup RKM servers in the `RKM.conf` file. If an IBM Spectrum Scale node cannot retrieve a master encryption key from the main RKM server, it tries each backup server in the list until it either retrieves a key or exhausts the list.

Note: This command does not do any operation on backup RKM servers. The system administrator is responsible for maintaining replication across all of the backup servers.

--distribute | --nodistribute

--distribute
Attempts to arrange the list of RKM server names (main RKM server and backup RKM servers) in the `RKM.conf` file in a different order on each node so that each node connects with the servers in a different order. This option provides some performance advantage in retrieving MEKs.

--nodistribute
Does not attempt to arrange the list of backup RKM server names in the `RKM.conf` file.

--timeout *ConnectionTimeout*
Sets the connection timeout in seconds for retrieving an MEK from an RKM server. The valid range is 1 - 120 seconds. The default is 60 seconds.

--retry *ConnectionAttempts*
Sets the number of attempts to retry a connection to an RKM server. The valid range is 1 - 10 retries. The default is three retries.

--interval *Microseconds*
Specifies the number of microseconds to wait between connection retries. The valid range is 1 - 1000000000. The default is 10000 (0.1 seconds).

show
Displays information about all the RKM stanzas in the `RKM.conf` file of the node.

Exit status

0 Successful completion.

Nonzero
A failure occurred.

Security

You must have root authority to run the **mmkeyserv** 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 the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

Examples

Examples 1-5 illustrate the steps in configuring an RKM server and a key client and generating an encryption key:

1. The following command makes an RKM server known to an IBM Spectrum Scale cluster. The name `keyserver01` is the host name of the ISKLM server:

```
# mmkeyserv server add keyserver01
Enter password for the RKM server keyserver01:
The security certificate(s) from keyserver01.gpfs.net must be accepted to continue.  View the
certificate(s) to determine whether you want to trust the certifying authority.
Do you want to view or trust the certificate(s)? (view/yes/no) view

Serial number:          01022a8adf20f3
SHA-256 digest:         2ca4a48a3038f37d430162be8827d91eb584e98f5b3809047ef4a1c72e15fc4c
Signature:              7f0312e7be18efd72c9d8f37dbb832724859ba4bb5827c230e2161473e0753b367ed49d9935
05bd23858541475de8e021e0930725abbd3d25b71edc8fc3de20b7c2db5cd4e865f41c7c410c1d710acf222e1c45189108e
40568ddcbeb21094264da60a1d96711015a7951eb2655363309d790ab44ee7b26adf8385e2c210b8268c5aede5f82f26855
4a6fc22ece6efeee2a6264706e71416a0dbe8c39ceacd86054d7cc34dda4fffea4605c037d32129055610821af85dd9819a
4d7e4baa70c51addcda720d33bc9f8bbde6d292c028b2f525a0275ebea968c26f8f0c4b604719ae3b04e71ed7a8188cd6ad
f68764374b29c91df3d101a941bf8b7189485ad72
Signature algorithm:    SHA256WithRSASignature
Key size:               2048
Issuer:                 C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=c40bbc1xn3.gpfs.net
Subject:                C=US, O=IBM, OU=SKLMNode, SKLMCell, CN=c40bbc1xn3.gpfs.net

Serial number:          01022a24475466
SHA-256 digest:         077c3b53c5046aa893b760c11cca3a993efbc729479771e03791f9ed4f716879
Signature:              227b5befe89f2e55ef628da6b50db1ab842095a54e1505655e3d95fee753a7f7554868aa79b
294c503dc34562cf69c2a2012879675883896856 5c0812c4aedbb0543d396646a269c02bf4c5ce5acba4409a10effbd47ca
38ce492698e2dcdc8390b9ae3f4a47c23ee3045ff0145218668f35a63edac68201789ed0db6e5c170f5c6db49769f0b4c9a
5f208746e4342294c447793ed087fa0ac762588faf420febeb3fca411e4e725bd46476e1f9f44759a696573af5dbbc95532
```

```
18c7083c80440f2e542bf56cc5cc18156cce05efd6c2e5fea2b886c5c1e262c10af18b13ccf38c3533ba025b97bbe62f271
545b2ab5c1f50c1dca45ce504dfcfc257362e9b43
Signature algorithm:    SHA256WithRSASignature
Key size:               2048
Issuer:                 C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=c40bbc1xn3.gpfs.net
Subject:                C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=c40bbc1xn3.gpfs.net

Do you trust the certificate(s) above? (yes/no) yes
To display all RKM servers information, enter:
c34f2n03:~ # mmkeyserv server show
keyserver01
        Type:               ISKLM
        Hostname:           keyserver01.gpfs.net
        User ID:            SKLMAdmin
        REST port:          9080
        Label:              1_keyserver01
        NIST:               on
        FIPS1402:           off
        Backup Key Servers:
        Distribute:         yes
        Retrieval Timeout:  120
        Retrieval Retry:    3
        Retrieval Interval: 10000
```

The following command displays information about all RKM servers that are known to the cluster. At the moment, the only one is `keyserver01`:

```
# mmkeyserv server show
keyserver01
        Type:               ISKLM
        Hostname:           keyserver01.gpfs.net
        User ID:            SKLMAdmin
        REST port:          9080
        Label:              1_keyserver01
        NIST:               on
        FIPS1402:           off
        Backup Key Servers:
        Distribute:         yes
        Retrieval Timeout:  120
        Retrieval Retry:    3
        Retrieval Interval: 10000
```

2. The following command creates a tenant in the server that you defined in Example 1, `keyserver01`. The name of the tenant is `devG1`:

```
# mmkeyserv tenant add devG1 --server keyserver01
Enter password for the RKM server keyserver01:
```

The following command displays all the current tenants of `keyserver01`:

```
# mmkeyserv tenant show
devG1
        Key Server:         keyserver01.gpfs.net
        Registered Client:  (none)
```

3. The following command adds a key client to the tenant that you created in Example 2. The command does not specify password files for the server and the new keystore, so the command prompts for the passwords. The name of the key client is `c34f2n03Client1`:

```
# mmkeyserv client create c34f2n03Client1 --server keyserver01
Enter password for the RKM server keyserver01:
Create a pass phrase for keystore:
Confirm your pass phrase:
```

The following command displays information about all the key clients on the RKM server:

```
# mmkeyserv client show
c34f2n03Client1
        Label:              c34f2n03Client1
        Key Server:         keyserver01.gpfs.net
        Tenants:            (none)
```

4. The following command registers the key client from Example 3 to the tenant from Example 2. To ensure uniqueness in RKM IDs, it is a good practice to create the RKM ID name by combining the names of the RKM server and the tenant. However, the RKM ID cannot be longer than 21 characters. In this example the RKM ID is `keyserver01_devG1`:

```
# mmkeyserv client register c34f2n03Client1 --tenant devG1 --rkm-id keyserver01_devG1
Enter password for the RKM server :

mmkeyserv: [I] Client currently does not have access to the key.  Continue the registration process ...
mmkeyserv: Successfully accepted client certificate
```

   The following two commands now show that key client **c34f2n03Client1** is registered to tenant **devG1**:

```
# mmkeyserv tenant show
devG1
        Key Server:         keyserver01.gpfs.net
        Registered Client:  c34f2n03Client1

# mmkeyserv client show
c34f2n03Client1
        Label:              c34f2n03Client1
        Key Server:         keyserver01.gpfs.net
        Tenants:            devG1
```

   The following command shows the contents of the new RKM stanza that was added to the `RKM.conf` file:

```
# mmkeyserv rkm show
keyserver01_devG1 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c34f2n03Client1.1.p12
  passphrase = pw4c34f2n03Client1
  clientCertLabel = c34f2n03Client1
  tenantName = devG1
}
```

   You can also show the contents of the `RKM.conf` file by routing the contents of the file to the console:

```
# cat /var/mmfs/ssl/keyServ/RKM.conf
keyserver01_devG1 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c34f2n03Client1.1.p12
  passphrase = pw4c34f2n03Client1
  clientCertLabel = c34f2n03Client1
  tenantName = devG1
}
```

5. The following example creates an encryption key in the tenant from Example 4. In the third line, the command displays the new encryption key (`KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1`):

```
# mmkeyserv key create --server keyserver01.gpfs.net --tenant devG1
Enter password for the RKM server keyserver01.gpfs.net:
KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1
```

See also

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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 163
- "mmcrfileset command" on page 223
- "mmdelfileset command" on page 271
- "mmlsfileset command" on page 365
- "mmunlinkfileset command" on page 524

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.

-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.ea1:             "value1"
```

See also

- "mmchattr command" on page 113

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 10
- "mmdelcallback command" on page 265

Location

```
/usr/lpp/mmfs/bin
```

mmlscluster command

Displays the current configuration information for a GPFS cluster.

Synopsis

mmlscluster [**--cnfs**] [**--ces**] [--cloud-gateway]

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.

--cloud-gateway
Displays information about Transparent Cloud Tiering 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. 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
   ```

2. To display the configuration information about the Transparent Cloud Tiering nodes, issue this command:

   ```
   mmlscluster --cloud-gateway
   ```

   The system displays output similar to this:

   ```
   GPFS cluster information
   ========================
     GPFS cluster name:         c350f1u1b11
     GPFS cluster id:           9364209917238477017

    Node  Daemon node name              Cloud node type
   -------------------------------------------------
      1   c350f1u1b11                    Cloud-Gateway
      2   c350f8u17.pk.labs.ibm.com      Cloud-Gateway
      3   c350f8u18.pk.labs.ibm.com      Cloud-Gateway
   ```

See also

- “mmaddnode command” on page 29
- “mmchcluster command” on page 119
- “mmcrcluster command” on page 218
- “mmdelnode command” on page 276

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 4.2.0.0
| 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 119
- "mmchconfig command" on page 123
- "mmcrcluster command" on page 218

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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.

Examples

1. To display the current state of **gpfs2nsd**, issue this command:

   ```
   mmlsdisk /dev/fs0 -d gpfs2nsd
   ```

   The system displays information similar to:

   ```
   disk      driver  sector failure holds    holds                     storage
   name      type      size   group metadata data  status  availability 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      driver sector failure holds    holds                            storage
   name      type     size   group metadata data  status availability disk id 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     driver sector failure holds   holds                                storage
   name     type   size   group  metadata data  status     availability  disk id 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     IO 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     IO performed on node     Device             Availability
------------  -----------------------  -----------------  ------------
gpfs7nsd      localhost                /dev/hdisk12       up
gpfs10nsd     k5n88.kgn.ibm.com             -             up
gpfs4nsd      localhost                /dev/hdisk10       up
```

See also

- "mmadddisk command" on page 23
- "mmchdisk command" on page 151
- "mmdeldisk command" on page 266
- "mmrpldisk command" on page 485

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: 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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 163
- “mmcrfileset command” on page 223
- “mmdelfileset command” on page 271
- “mmlinkfileset command” on page 350
- “mmunlinkfileset command” on page 524

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 DMAPI 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.

--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.

Note: The command treats the following conditions as failures:

- The file system that you specified was not found.
- You specified **all**, **all_local**, or **all_remote** and no file systems were found.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 229
- "mmchfs command" on page 169
- "mmdelfs command" on page 274

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 IBM Spectrum Scale license designation, see *IBM Spectrum Scale license designation* in *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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
This node runs IBM Spectrum Scale Advanced Edition
```

mmlslicense

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
This node runs IBM Spectrum Scale Advanced Edition
```

See also

- "mmchlicense command" on page 175

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 (kolt2g_r1b42)
   ```

See also

- "mmchmgr command" on page 178

Location

```
/usr/lpp/mmfs/bin
```

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 **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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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

- "mmmount command" on page 400
- "mmumount command" on page 521

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 all node classes defined in the system, issue this command:

   ```
   mmlsnodeclass --all
   ```

   The system displays information similar to:

```
Node Class Name       Members
--------------------- -----------------------------------------------------------
aixNodes
all                   node03,node01,node02,node04,nodensd01,nodensd02
cesNodes              node01,node02,node04
clientLicense
clientNodes           node03,node02,node04,nodensd01,nodensd02
cnfsNodes
disabledCnfsNodes
enabledCnfsNodes
linuxNodes            node03,node01,node02,node04,nodensd01,nodensd02
managerNodes          node01
nonAixNodes           node03,node01,node02,node04,nodensd01,nodensd02
nonCesNodes           node03,nodensd01,nodensd02
nonCnfsNodes          node03,node01,node02,node04,nodensd01,nodensd02
nonLinuxNodes
nonNsdNodes           node003,node01,node02,node04
nonQuorumNodes
nonWindowsNodes       node03,node01,node02,node04,nodensd01,nodensd02
nsdNodes              nodensd01,nodensd02
quorumNodes           node03,node01,node02,node04,nodensd01,nodensd02
serverLicense         node03,node01,node02,node04,nodensd01,nodensd02
windowsNodes
siteA                 linuxNodes
object_database_node  node01
object_singleton_node node01
```

3. 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 185
- "mmcrnodeclass command" on page 239
- "mmdelnodeclass command" on page 279

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 *DiskName*s, 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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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           sdcnsd       c5n94g,c5n96g
    fs2           sddnsd       c5n94g,c5n96g
    fs2           sdensd       c5n94g,c5n96g
    fs2           sdgnsd       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:

```
mmlsnsd -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 **mmlsnsd -X** with **mmchconfig usePersistentReserve=yes**.

```
 Disk name    NSD volume ID      Device         Devtype  Node name                Remarks
-------------------------------------------------------------------------------------------------
 nsd0         947AC0A84F5FB55C   /dev/dm-0      dmm      c13c1apv12.gpfs.net      server node,pr=yes
 nsd0         947AC0A84F5FB55C   /dev/dm-19     dmm      c13c1apv13.gpfs.net      server node,pr=yes
 nsd0         947AC0A84F5FB55C   /dev/dm-6      dmm      c13c1apv14.gpfs.net      server node,pr=yes
 nsd0         947AC0A84F5FB55C   /dev/dm-15     dmm      c13c1apv16.gpfs.net      server node,pr=yes
 nsd1         947AC0A84F5FB564   /dev/dm-1      dmm      c13c1apv12.gpfs.net      server node,pr=yes
 nsd1         947AC0A84F5FB564   /dev/dm-4      dmm      c13c1apv13.gpfs.net      server node,pr=yes
 nsd1         947AC0A84F5FB564   /dev/dm-9      dmm      c13c1apv14.gpfs.net      server node,pr=yes
 nsd1         947AC0A84F5FB564   /dev/dm-13     dmm      c13c1apv16.gpfs.net      server node,pr=yes
```

See also

- "mmcrnsd command" on page 241
- "mmdelnsd command" on page 281

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 *Information Lifecycle Management* in *IBM Spectrum Scale: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 56
- "mmchpolicy command" on page 191

Location

/usr/lpp/mmfs/bin

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 fs1 p1 -L
   ```

   The system displays information similar to this:

```
|     Pool:
|       name               = p1
|       poolID             = 65537
|       blockSize          = 256 KB
|       usage              = dataOnly
|       maxDiskSize        = 497 GB
|       layoutMap          = cluster
|       allowWriteAffinity = no
|       writeAffinityDepth = 0
|       blockGroupFactor   = 1
```

See also

- "mmlsattr command" on page 352
- "mmlscallback command" on page 355
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359
- "mmlsdisk command" on page 361
- "mmlspolicy command" on page 384
- "mmlsquota command" on page 391
- "mmlssnapshot command" on page 395

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

With the **mmlsqos** command, you can display the consumption of I/O operations by processes that access designated storage pools. With the **mmchqos** command, you can regulate I/O access to a specified storage pool by allocating shares of I/O operations to two QoS classes:

maintenance
The default QoS class for some I/O intensive, potentially long-running GPFS commands, such as **mmbackup**, **mmrestore**

other The default QoS class for all other processes.

A third class, **misc**, is used to count the IOPS that some critical file system processes consume. You cannot assign IOPS to this class, but its count of IOPS is displayed in the output of the **mmlsqos** 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 Guide*.

When the file system is mounted, the command displays information about the QoS classes of both explicitly named pools and unnamed pools. *Unnamed 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 Guide*.

Analyzing the output from mmlsqos

The **mmlsqos** command always shows the first three lines of output:

QOS config::
Indicates whether QoS is actively regulating I/O consumption (enabled) or is quiescent (disabled).

QOS values::
Displays, for each storage pool that you configured, the name of the storage pool and the IOPS that you assigned to the **other** class and the **maintenance** class. In the following example fragment, the command shows that the system storage pool is configured with the value of **inf** for both QoS classes:

```
QOS values:: pool=system,other=inf,maintenance/all_local=inf
```

The qualifier `/all_local` after `maintenance` indicates that the maintenance IOPS are applied to all the files systems owned by the cluster. This value is the default for the `maintenance` class.

QOS status::
Indicates whether QoS is regulating the consumption of IOPS ("throttling") and also whether QoS is recording ("monitoring") the consumption of IOPS of each storage pool.

The following sample output is complete:

```
# mmlsqos fs --seconds 30
QOS config::     enabled
QOS values::     pool=system,other=inf,maintenance/all_local=inf:pool=fpodata,other=inf,maintenance/all_local=inf
QOS status::     throttling active, monitoring active
=== for pool fpodata
01:31:45  misc iops=11 ioql=0.016539 qsdl=1.2e-06 et=5
=== for pool system
01:31:45  misc iops=8.2 ioql=0.013774 qsdl=2e-06 et=5
```

The command `mmlsqos fs0 --seconds 30` requests a display of I/O performance values for all QoS pools over the previous 30 seconds. Because the parameters **--sum_classes** and **--sum_nodes** are missing, the command also requests I/O performance for each storage pool separately and summed across all the nodes of the cluster.

The information that is displayed for the two configured pools, `fpodata` and `system`, indicates that IOPS occurred only for processes in the **misc** class. The meaning of the categories in each line is 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.

ioql= 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.

qsdl= 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 `Qsdl` 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 + Qsdl)/Iops*. For a system that is running IO-intensive applications, you can interpret the value *(Ioql + Qsdl)* 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 196
- *Setting the Quality of Service for I/O operations (QoS)* in the *IBM Spectrum Scale: Administration Guide*.

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

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.

For more information, see *Listing quotas* in *IBM Spectrum Scale: Administration Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

   ```
                            Block Limits                |     File Limits
   Filesystem type 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:

   ```
                            Block Limits                    |     File Limits
                                                  in                              in
   Filesystem Fileset type   KB quota  limit doubt grace | files quota limit doubt grace  Remarks
   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:

   ```
           Default Block Limits(KB)                 | Default File Limits
   Filesystem Fileset    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
   ```

```
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:

```
                     Block Limits                                    |     File Limits
Filesystem type      blocks   quota   limit   in_doubt    grace |  files   quota    limit  in_doubt   grace  Remarks
fs1        FILESET   89.25G    100G    200G     89.99M     none |  13729    4000     5000         0  7 days
```

See also

- "mmcheckquota command" on page 159
- "mmdefedquota command" on page 251
- "mmdefquotaoff command" on page 254
- "mmdefquotaon command" on page 257
- "mmedquota command" on page 298
- "mmrepquota command" on page 463
- "mmquotaon command" on page 455
- "mmquotaoff command" on page 453

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.

--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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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

Note: Ensure that the snapshot name does not include a colon (:).

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
```

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 246
- "mmdelsnapshot command" on page 283
- "mmrestorefs command" on page 471
- "mmsnapdir command" on page 511

Location

```
/usr/lpp/mmfs/bin
```

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 123
- "mmchfs command" on page 169

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 Guide*.

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* in *IBM Spectrum Scale: Administration Guide*.

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 the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

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 521
- "mmlsmount command" on page 377

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]
                         [--nfsremove {Client[,Client...]}]
                         [--nfschange ClientOptions]
                         [--nfsposition {PositionIndex | Client}]
```

or

```
mmnfs export list [--nfsdefs Path] [-Y]
```

or

```
mmnfs export load ExportCFGFile
```

or

```
mmnfs configuration list [--exportdefs] [-Y]
```

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 *Path* so that clients can connect to it. If there is already an existing export for the *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 might lead to serious issues in data consistency. Be very cautious when creating and using nested exports.

If there is a need to have nested exports (such as `/path/to/folder` and `/path/to/folder/inside/subfolder`), NFSv4 client that mounts the parent (`/path/to/folder`) export will not be able to see the child export subtree (`/path/to/folder/subfolder`) unless the same client is explicitly allowed to access the child export as well. This is okay as long as the client uses only NFSv4 mounts.

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=RO,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. To avoid incorrect parsing by the interpreter, quote the argument list. 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: The **mmnfs export change** will restart NFS services on all CES nodes on which the NFS server is currently running.

--nfsadd *ClientOptions*
Adds a new client declaration for the specified *Path*. *ClientOptions* can be a list of one or more client definitions. To avoid incorrect parsing by the interpreter, quote the argument list. 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*.

--nfsremove {*Client***[,***Client***...]}**
Removes the NFS client specified by *Client* from the export configuration for the *Path*. *Client* can be a single client specifier, or a comma-separated list of client specifiers. If a client is the only one within a client declaration section in the configuration file, then this client section is removed.

Note: When the last client within a client declaration is removed, then the export section is also removed.

--nfschange *ClientOptions*
Modifies an existing client declaration for the specified *Path*. *ClientOptions* can be a list of one or more client definitions. To avoid incorrect parsing by the interpreter, quote the argument list. 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*. If a client specifier declared in *ClientOptions* matches a client declaration section, which has further NFS clients assigned, then this section is split into a section containing that client specifier together with the modified attributes, and a section with the remaining client specifiers and their original attributes.

The **--nfsposition** option can be used to locate this modified entry.

--nfsposition {*PositionIndex* **|** *Client***}**
This option can be used only together with **--nfsadd** or **--nfschange**. It reorders the client declaration section within the export declaration file. The sequence of client entries can be important when there are different client sections affecting the same NFS client. For example, NFS position can be important when defining export attributes for a specific NFS client, and '*' for all NFS clients, given the same export path.

PositionIndex
This must be an unsigned numeric value, and it indicates the absolute sequence number after reordering the client declarations. The indexing is zero-based. 0 means the top of the list, 1 means the second entry, and so on. Any number higher than the number of existing entries indicates the end-of-list position.

Client
The reference client specifier to indicate the new position of the current client entry. The modified client entry will be placed at the location where the *Client* reference declaration is, and that section is ordered below next to the modified client section.

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.

--nfsdefs *String* **-Y**
Lists the export configuration details, in machine readable format, for all export paths that contain the specified *String*.

-Y 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 **-Y** option to create the machine readable output.

--nfsdefs *String* **-Y**
Lists the export configuration details, in machine readable format, for all export paths that contain the specified String.

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, hostnames, netgroups, or * for all. A netgroup name must not start with a numeric character and otherwise must only contain only underscores ("[a-zA-Z_][0-9a-zA-Z_]*"). 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.

-Y 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 **-Y** 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.

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.

LEASE_LIFETIME

Allowed values are between 0 and 120. The default value is 60.

DOMAINNAME

String. The default value is the "host's fully-qualified DNS domain name".

IDMAPD_DOMAIN

String. Domain in the ID Mapd configuration. The default value is the host's fully-qualified DNS domain name.

LOCAL REALMS
String. Local-Realms in the ID Mapd configuration. The default value is the host's default realm name.

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. 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To create an NFS export (using a netgroup), issue this command:

```
mmnfs export add /mnt/gpfs0/netgrouppath \
--client "@netgroup(Access_Type=RO,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 DefaultDelegations Manage_Gids NFS_Commit
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
/gpfs/fs1/export_1   none      192.0.2.8    RW        3,4      TCP        ROOT_SQUASH   -2            -2        SYS     FALSE          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 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 Protocols Transports Squash         Anonymous Anonymous SecType Privileged  Default    Manage_Gids NFS_Commit
                                          _Type                                        _uid      _gid              Port                  Delegation
-------------------------------------------------------------------------------------------------------------------------------------------------------------------
/gpfs/fs1/export_1   none      192.0.2.8      RW       3,4       TCP     ROOT_SQUASH      -2       -2        SYS     FALSE    none        FALSE       FALSE
/gpfs/fs1/export_1   none      198.51.100.10  MDONLY   3,4       TCP     allsquash        -2       -2        SYS     FALSE    none        FALSE       FALSE
/gpfs/fs1/export_1   none      192.0.2.0/20   RW       3,4       TCP     no_root_squash   -2       -2        SYS     FALSE    none        FALSE       FALSE
```

Note: The **mmnfs export change** will restart NFS services on all CES nodes on which the NFS server is currently running.

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:

```
192.168.80.131:  Redirecting to /bin/systemctl stop nfs-ganesha.service
192.168.80.135:  Redirecting to /bin/systemctl stop nfs-ganesha.service
192.168.80.131:  Redirecting to /bin/systemctl start nfs-ganesha.service
192.168.80.135:  Redirecting to /bin/systemctl start nfs-ganesha.service
NFS Configuration successfully changed. NFS server restarted on all NFS
nodes on which NFS server is running.
```

Note: This command removes a single client definition, for the IP "1.2.3.1", from the NFS export. If this is the last client definition for the export, then the export is also removed.

5. To modify an NFS export, issue this command:

```
mmnfs export change  /mnt/gpfs0/p1 \
--nfschange "203.0.113.2(Access_Type=RO)" --nfsposition 0
```

The system displays output similar to this:

```
192.168.80.131: Redirecting to /bin/systemctl stop nfs-ganesha.service
192.168.80.135: Redirecting to /bin/systemctl stop nfs-ganesha.service
192.168.80.131: Redirecting to /bin/systemctl start nfs-ganesha.service
192.168.80.135: Redirecting to /bin/systemctl start nfs-ganesha.service
NFS Configuration successfully changed. NFS server restarted on all NFS
nodes on which NFS server is running.
```

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_ Protocols Transports Squash      Anonymous Anonymous  SecType  Privileged Default   MANAGE  NFS
                                       Type                                     _uid      _gid                Port    Delegations  _Gids _Commit
-------------------------------------------------------------------------------------------------------------------------------------------------
/mnt/gpfs0/p1  none        203.0.113.2 RO      3,4       TCP        ROOT_SQUASH  -2        -2         SYS      FALSE      none     FALSE    FALSE
/mnt/gpfs0/p1  none        *           RW      3,4       TCP        ROOT_SQUASH  -2        -2         SYS      FALSE      none     FALSE    FALSE
/mnt/gpfs0/p1  none        203.0.113.1 RO      3,4       TCP        ROOT_SQUASH  -2        -2         SYS      FALSE      none     FALSE    FALSE
```

8. To list the 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
SHORT_FILE_HANDLE: FALSE
LEASE_LIFETIME: 60
DOMAINNAME: VIRTUAL1.COM
DELEGATIONS: Disabled
==========================

STATD Configuration
==========================
STATD_PORT: 0
==========================

CacheInode Configuration
==========================
ENTRIES_HWMARK: 1500000
==========================

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
==========================
LOG_LEVEL: EVENT
==========================

Idmapd Configuration
==========================
LOCAL-REALMS: LOCALREALM
DOMAIN: LOCALDOMAIN
==========================
```

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 Configuration successfully changed. NFS server restarted on all NFS nodes
on which NFS server is running.
```

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 96
- "mmchconfig command" on page 123
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359
- "mmobj command" on page 414
- "mmsmb command" on page 500
- "mmuserauth command" on page 527

Location

```
/usr/lpp/mmfs/bin
```

mmnsddiscover command

Rediscovers 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 Rediscovers 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 Guide*.

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

more information, see *Requirements for administering a GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 188
- "mmcrnsd command" on page 241
- "mmdelnsd command" on page 281
- "mmlsnsd command" on page 381

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] [--ces-group CESGroup]
                   {{{--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-encryption --encryption-keyfile EncryptionKeyFileName [--force-rule-append]]}
                   [--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 enable
```

| or

```
| mmobj file-access disable[--objectizer]
| mmobj file-access objectize {--object-path ObjectPath | --storage-policy
| PolicyName [--account-name AccountName [ --container-name ContainerName [--object-name
| ObjectName]]]} [-N | --node NodeName]
```

| or

```
mmobj multiregion list
```

or

mmobj multiregion enable

or

mmobj multiregion export --region-file *RegionFile*

or

mmobj multiregion import --region-file *RegionFile*

or

mmobj multiregion remove --region-number *RegionNumber* **--force**

or

mmobj s3 enable | **disable** | **list**

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.

If there is an existing object authentication configured, use the **mmuserauth service remove --data-access-method object** command to remove the object authentication. Then, use the **mmces service disable OBJ** command to perform the 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.

In the unified file and object access environment, the ibmobjectizer service can be used for making files created from the file interfaces, such as POSIX, NFS, and CIFS accessible through object interfaces, such as curl and SWIFT. The ibmobjectizer service runs periodically and makes files available for the object interface. The `file-access` option of the **mmobj** command can be used to enable and disable the file-access capability and the ibmobjectizer service. The ibmobjectizer service runs on the CES IP address with the object_singleton_node attribute.

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.

--ces-group
Specifies the CES group that contains the IP addresses to be used for the swift ring files. This allows you to specify a subset of the overall collection of CES IP addresses to be used by the object protocol.

--enable-s3
Sets the `s3` capability (Amazon S3 API support) to `true`. By default, S3 API 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.

--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.

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 treaded as *.

--enable-encryption
Enables an encryption policy.

--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.

--encryption-keyfile ***EncryptionKeyFileName***

Specifies the encryption key file.

--force-rule-append

Specifies whether to append and establish the rule if other rules are already existing

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

Manages file-access capability, ibmobjectizer service and object access for files (objectizes) in a unified file and object access environment.

enable

Enables the file-access capability and the ibmobjectizer service.

If the file access capability is already enabled and the ibmobjectizer service is stopped, this option starts the ibmobjectizer service

disable
Disables the file-access capability and the ibmobjectizer service.

--objectizer
This option stops only the ibmobjectizer service, it does not disable the file-access capabilities.

objectize
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 that you want to enable for access through the object interface. If a fully-qualified path to a directory is specified, 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 ***AcccountName***
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.

-N | --node ***NodeName***
The node on which the command is to be executed. Optional. If this parameter is not specified, the command is executed on 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.

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.

s3 Enables and disables the S3 API without manually changing the configuration.

enable
Enables the S3 API.

disable
Disables the S3 API.

list
Verifies if the S3 API has been enabled or disabled.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. To specify configuration of Object protocol service with local keystone and S3 enabled, run the following 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, run the following 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, run the following 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, run the following 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, run 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"
11751510260 Test19                    obj_Test19       /ibm/cesSharedRoot/obj_Test19                              regions="1"
```

7. To change the default storage policy, run the following command:

```
mmobj policy change sof-policy --default
```

   The system displays sof-policy as the default storage policy.

8. To enable object access for an account, run the following 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
```

9. To enable object access for a container, run the following command:

```
mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1
```

10. To enable object access on a file while specifying a storage policy, run the following command:

```
mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1 --object-name file1.txt
```

11. To enable object access on a file, run the following command:

```
mmobj file-access --object-path /ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_12345/container1/file1.txt
```

12. To list the information about a region, run the following 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
```

13. To set up the initial multi-region environment on the first region, run the following command:

```
mmobj multiregion enable
```

The system displays output similar to this:

```
mmobj multiregion: Multi-region support is enabled in this cluster.  Region number: 1
```

14. To export multi-region data for use by other clusters to join multi-region, run the following 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
```

15. To import the specified multi-region configuration environment created by the export command into a region, run the following 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
```

16. To remove a region designated by region number 2 from a multi-region environment and to remove all configuration information of the specified region, run the following 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.
```

17. To create a policy with no force add where only the default GPFS policy rule is established, run the following command:

```
mmobj policy create enc-policy-1 -i 10000 --enable-encryption --encryption-keyfile /root/enc/enc.key
```

The system creates the policy, adds and establishes the rule within the GPFS policy rules, and displays the following output:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_enc-policy-1
[I] Creating GPFS Encryption Rule
```

```
[I]  The following new encryption rule has been stored to file:/var/mmfs/ces/policyencryption.rule
 and is established within the  GPFS policies.

/* begin of the encryption rule for fileset obj_enc-policy-1 */
rule 'enc-1_enc-policy-1' set encryption 'encryption_enc-policy-1'
for fileset('obj_enc-policy-1')
where name like '%'

rule 'enc-2_enc-policy-1' encryption 'encryption_enc-policy-1'
is  ALGO 'DEFAULTNISTSP800131A'
KEYS('KEY-2712784a-d1ee-4c86-bf97-b88f918cbd12:sklm')
/* end of the encryption rule for fileset obj_enc-policy-1 */

[I] Creating new unique index and building the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

18. To create a policy with force add, run the following command:

```
mmobj policy create enc-policy-2 -i 10000 --enable-encryption --encryption-keyfile /root/enc/enc.key
 --force-rule-add
```

The system creates the policy, adds and establishes the rule within GPFS policy rules, and displays the following output:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_enc-policy-2
[I] Creating GPFS Encryption Rule
[I]  The following new encryption rule has been stored to file:/var/mmfs/ces/policyencryption.rule
 and is established within the  GPFS policies.

/* begin of the encryption rule for fileset obj_enc-policy-2 */
rule 'enc-1_enc-policy-2' set encryption 'encryption_enc-policy-2'
for fileset('obj_enc-policy-2')
where name like '%'

rule 'enc-2_enc-policy-2' encryption 'encryption_enc-policy-2' is
ALGO 'DEFAULTNISTSP800131A'
KEYS('KEY-2712784a-d1ee-4c86-bf97-b88f918cbd12:sklm')
/* end of the encryption rule for fileset obj_enc-policy-2 */

[I] Creating new unique index and building the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

19. To create a policy with no force add, run the following command:

```
mmobj policy create enc-policy-3 -i 10000 --enable-encryption --encryption-keyfile /root/enc/enc.key
```

The system creates the policy, adds the rule but does not establish the rule within the GPFS policy rules, and displays the following output:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_enc-policy-3
[I] Creating GPFS Encryption Rule
[I]  The following new encryption rule has been stored to file:/var/mmfs/ces/policyencryption.rule
 but is not established within  the GPFS policies.

/* begin of the encryption rule for fileset obj_enc-policy-3 */
rule 'enc-1_enc-policy-3' set encryption 'encryption_enc-policy-3'
for fileset('obj_enc-policy-3')
where name like '%'

rule 'enc-2_enc-policy-3' encryption 'encryption_enc-policy-3' is
ALGO 'DEFAULTNISTSP800131A'
KEYS('KEY-2712784a-d1ee-4c86-bf97-b88f918cbd12:sklm')
/* end of the encryption rule for fileset obj_enc-policy-3 */

[I] Creating new unique index and building the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

20. To enable the ibmobjectizer service, run the following command:
```
mmobj file-access enable
```
The system enables the file-access capability and starts the ibmobjectizer service.
21. To disable the ibmobjectizer service, run the following command:
```
mmobj file-access disable
```
The system disables the file-access capability and stops the ibmobjectizer service.
22. To use the disable --objectizer-daemon parameter, run the following command:
```
mmobj file-access disable --objectizer-daemon
```
The system disables the ibmobjectizer service.
23. To use the object-path parameter, run the following command:
```
mmobj file-access objectize --object-path /ibm/cesSharedRoot/fileset1/Auth_12345/container1/file1.txt
```
The system objectizes file1.txt at /ibm/cesSharedRoot/fileset1/Auth_12345/container1/ and enables it for access through the object interface:
```
Loading objectization configuration from CCR
Fetching storage policy details
Performing objectization
Objectization complete
```
24. To use the storage-policy parameter for file objectization, run the following command:
```
mmobj file-access objectize --storage-policy sof_policy --account-name admin
 --container-name container1 --object-name file1.txt
```
The system objectizes file1.txt from the container named container1:
```
Loading objectization configuration from CCR
Fetching storage policy details
Performing objectization
Objectization complete
```
25. To use the node parameter for running the command on a remote node, run the following command:
```
mmobj file-access objectize --node gpfs_node1 --storage-policy sof_policy --account-name admin
```
The command is run on the gpfs_node1 node and all the files from all containers within the admin account are objectized. If **--node** or **-N** is not specified, the **mmobj file-access objecitze** command checks if the current node is CES node or not. If the current node is a CES node, the command is executed on the current node. If the current node is not a CES node, the command is executed on a random remote CES node:
```
Loading objectization configuration from CCR
Fetching storage policy details
Performing objectization
Objectization complete
```

See also

- "mmces command" on page 96
- "mmchconfig command" on page 123
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359
- "mmnfs command" on page 402
- "mmsmb command" on page 500
- "mmuserauth command" on page 527

Location

/usr/lpp/mmfs/bin

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"
```

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 Guide*.
- --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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
{
```

```
        name = "GPFSDiskCap"
        period = 0
},
{
        name = "NFSIO"
        period = 0
        proxyCmd = "/opt/IBM/zimon/GaneshaProxy"
        restrict = "cesNodes"
        type = "Generic"
}
smbstat = ""
```

5. To display the currently active performance monitoring configuration, issue the command:

```
# mmperfmon 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 = "GPFSPDDisk"
        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
```

```
},
{
        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:

   ```
   mmperfmon 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:

   ```
   mmperfmon 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:

   ```
   mmperfmon 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
        passdb
        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
```

```
        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:

```
mmperfmon 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 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 Guide*.

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 Guide*.

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.

vio_s [f rg *RecoveryGroupName* **[da** *DeclusteredArray* **[v** *Vdisk***]]] [reset]**
Displays IBM Spectrum Scale RAID vdisk I/O statistics. For more information about IBM Spectrum Scale RAID, see *IBM Spectrum Scale RAID: Administration*.

vio_s_reset [f rg *RecoveryGroupName* **[da** *DeclusteredArray* **[v** *Vdisk***]]]**
Resets IBM Spectrum Scale RAID vdisk I/O statistics. For more information about IBM Spectrum Scale 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 Guide*.

-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 Guide*.
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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
   ```

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| check}
```

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 NFS, SMB, Object, or Networking operations. 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 this command, see the topic *CES tracing and debug data collection* in the *IBM Spectrum Scale: Problem Determination Guide*.

Parameters

options
Specifies one of the following trace options:

-d *Duration*
Specifies the trace duration in minutes. The default is **10**.

-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 names of the CES nodes where you want tracing to be done. The default is all the CES nodes. For more information, see the topic *Tips for using mmprotocoltrace* in the *IBM Spectrum Scale: Problem Determination Guide*.

-c *ClientIPs*
Specifies a comma-separated list of client IP addresses to trace. The CES nodes that you specified in the -N parameter will trace their connections with these clients. This parameter applies only to SMB traces and Network traces. For more information, see the topic *Tips for using mmprotocoltrace* in the *IBM Spectrum Scale: Problem Determination Guide*.

-f
Forces an action to occur. Affects the **clear** command.This parameter also disables prompt for smb and smbsyscalls.

-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.

check
Checks and performs all planned tracing actions specified on the node.

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:

nfs
Traces the NFS service.

smb
Traces the SMB service.

object
Traces the Object service.

network
Traces the Network service.

smbsyscalls
Collects the strace-logs for SMB.

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 Guide*.

Examples

1. To start an SMB trace, issue this command:

```
mmprotocoltrace start smb -c <clientIP>
```

   The system displays output similar to this:

```
Trace 'fcb7cb07-c45e-43f8-8f1f-2de50cf15062' created successfully for 'smb'
```

2. To view the status of the SMB trace, issue this command:

```
mmprotocoltrace status smb
```

   The system displays output similar to this:

```
Trace ID:     fcb7cb07-c45e-43f8-8f1f-2de50cf15062
State:        Active
User ID:      root
Protocol:     smb
Start Time:   10:57:43 04/03/2016
End Time:     11:07:43 04/03/2016
Client IPs:   10.0.100.42, 10.0.100.43
Origin Node:  ch-42.localnet.com
Syscall:      False
Syscall Only: False
Nodes:
      Node Name:      ch-41.localnet.com
      State:          ACTIVE
      Trace Location:  /tmp/mmfs/smb.20160304_105742.trc

      Node Name:      ch-42.localnet.com
      State:          ACTIVE
      Trace Location: /tmp/mmfs/smb.20160304_105742.trc

      Node Name:      ch-43.localnet.com
      State:          ACTIVE
      Trace Location: /tmp/mmfs/smb.20160304_105742.trc
```

3. To stop the SMB trace, issue this command:

```
mmprotocoltrace stop smb
```

   The system displays output similar to this:

```
Stopping traces
Trace 'fcb7cb07-c45e-43f8-8f1f-2de50cf15062' stopped for smb
Waiting for traces to complete
Waiting for node 'node1'
Waiting for node 'node2'
Waiting for node 'node3'
Finishing trace 'fcb7cb07-c45e-43f8-8f1f-2de50cf15062'
Successfully copied file from 'node1:/tmp/mmfs/smb.20160304_105742.trc'
```

```
Successfully copied file from 'node2:/tmp/mmfs/smb.20160304_105742.trc'
Successfully copied file from 'node3:/tmp/mmfs/smb.20160304_105742.trc'
Trace tar file has been written to '/tmp/mmfs/smb.trace.20160304_105845.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:

```
All traces cleared successfully
```

5. To trace the systemcalls for SMB, issue this command:

```
mmprotocoltrace smbsyscalls -c <clientIP>
```

The system displays output similar to this:

```
Trace '9cd534c9-be3c-4478-ba45-2e00acd4b544' created successfully for 'smb'
```

See also

- “mmaddcallback command” on page 10
- “mmces command” on page 96
- “mmchconfig command” on page 123
- “mmlscluster command” on page 357
- “mmlsconfig command” on page 359
- “mmnfs command” on page 402
- “mmsmb command” on page 500
- “mmuserauth command” on page 527

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. You can use this command only in a Single writer (SW) cache. You cannot use for Read only (RO), Independent writer (IW), or Local updates (LU) caches. Peer snapshots are not allowed on a Single writer (SW) cache that uses the NSD protocol for communicating with home.

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:
{*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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

1. 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.
```
2. 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
```
3. 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 37
- "mmafmctl command" on page 40
- "mmafmlocal command" on page 54
- "mmchattr command" on page 113
- "mmchconfig command" on page 123
- "mmchfileset command" on page 163
- "mmchfs command" on page 169
- "mmcrfileset command" on page 223
- "mmcrfs command" on page 229
- "mmlsconfig command" on page 359
- "mmlsfileset command" on page 365
- "mmlsfs command" on page 369

Location

`/usr/lpp/mmfs/bin`

mmputacl command

Sets the GPFS access control list for the specified file or directory.

Synopsis

mmputacl [**-d**] [**-i** *InFilename*] *Filename*

Availability

Available on all IBM Spectrum Scale editions. Available on AIX and Linux.

Description

Use the **mmputacl** 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 Guide*.

Any output from the **mmgetacl** command can be used as input to **mmputacl**. 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 **mmputacl** works for POSIX and NFS V4 ACLs:

```
Command       POSIX ACL                    NFS V4 ACL
-----------   --------------------------   ----------------------
mmputacl      Access ACL (Error if default   Stores the ACL (implies
               ACL is NFS V4 [1])             default as well)
mmputacl -d   Default ACL (Error if access   Error: NFS V4 ACL (has
               ACL is NFS V4 [1]              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 mmdelacl 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**), **mmputacl** may be restricted. The **mmputacl** command is not allowed to store an NFS V4 ACL if **-k posix** is in effect. The **mmputacl** 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 **mmputacl 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 **mmputacl -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::rwxc
group::rwx-
other::--x-
mask::rw-c
user:alpha:rwxc
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::rwxc
group::rwx-
other::--x-
mask::rw-c
user:alpha:rwxc
group:audit:rwx-
group:system:-w--
```

See also

- "mmeditacl command" on page 295
- "mmdelacl command" on page 263

- "mmgetacl command" on page 317

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

GPFS must be running on the node from which the **mmquotaoff** command is issued.

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 159
- “mmdefedquota command” on page 251
- “mmdefquotaoff command” on page 254
- “mmdefquotaon command” on page 257
- “mmedquota command” on page 298
- “mmlsquota command” on page 391
- “mmquotaon command” on page 455
- “mmrepquota command” on page 463

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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 159
- "mmdefedquota command" on page 251
- "mmdefquotaoff command" on page 254
- "mmdefquotaon command" on page 257
- "mmedquota command" on page 298
- "mmlsquota command" on page 391
- "mmquotaoff command" on page 453
- "mmrepquota command" on page 463

Location

```
/usr/lpp/mmfs/bin
```

mmremotecluster command

Manages information about remote GPFS clusters.

Synopsis

mmremotecluster add *RemoteClusterName* [**-n** *ContactNodes*] [**-k** *KeyFile*]

or

mmremotecluster update *RemoteClusterName* [**-C** *NewClusterName*] [**-n** *ContactNodes*] [**-k** *KeyFile*]

or

mmremotecluster delete {*RemoteClusterName* | **all**}

or

mmremotecluster show [*RemoteClusterName* | **all**]

Availability

Available on all IBM Spectrum Scale editions.

Description

The **mmremotecluster** 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 **mmremotecluster** 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 **mmremotecluster 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 **mmremotecluster** 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 **mmremotecluster update** command.

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 **mmremotecluster delete** or **mmremotecluster 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 **mmremotecluster** 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 **mmremotecluster** 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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.

   ```
   mmremotecluster add k164.kgn.ibm.com -n k164n02,k164n03 -k k164.id_rsa.pub
   ```

   The output is similar to this:

   ```
   mmremotecluster: 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**.

   ```
   mmremotecluster 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*.

3. 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.

   ```
   mmremotecluster update k164.kgn.ibm.com -n tcpPort=6667,k164n02,k164n01
   ```

   The output is similar to this:

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

   The **mmremotecluster show** command can then be used to see the changes.

   ```
   mmremotecluster 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*.

4. This command deletes information for remote cluster **k164.kgn.ibm.com** from the local cluster.

   ```
   mmremotecluster delete k164.kgn.ibm.com
   ```

   The output is similar to this:

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

See also

- "mmauth command" on page 67
- "mmremotefs command" on page 460

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: 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 **mmremotecluster** 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* in *IBM Spectrum Scale: Administration Guide*.

-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 **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.

--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.

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 Guide*.

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.fvtdomain.net /rgpfs1          rw               no         K
```

See also

- "mmauth command" on page 67
- "mmremotecluster command" on page 457

See also the topic about accessing GPFS file systems from other GPFS clusters in the *IBM Spectrum Scale: 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.

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.

For more information, see *Listing quotas* in *IBM Spectrum Scale: Administration Guide*.

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 the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

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:

```
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 159
- "mmdefedquota command" on page 251
- "mmdefquotaoff command" on page 254
- "mmdefquotaon command" on page 257
- "mmedquota command" on page 298
- "mmlsquota command" on page 391
- "mmquotaoff command" on page 453
- "mmquotaon command" on page 455

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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
   ```

```
  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 81
- "mmimgbackup command" on page 329
- "mmimgrestore command" on page 333

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 14 on page 472 and Table 15 on page 472 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: 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.

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 14. 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 15. 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 Guide*.

--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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
/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 246
- "mmdelsnapshot command" on page 283
- "mmlssnapshot command" on page 395
- "mmsnapdir command" on page 511

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 *Restriping a GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 mmrestorefs 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

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

This example illustrates restriping a file named `testfile0`. The following command confirms that the file is ill-placed:

```
mmlsattr -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:

```
mmlsattr -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 23
- "mmapplypolicy command" on page 56
- "mmchattr command" on page 113
- "mmchdisk command" on page 151
- "mmdeldisk command" on page 266
- "mmrpldisk command" on page 485
- "mmrestripefs command" on page 478

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 | -R | -c [--read-only]  | -p | -z}
                    [-N {Node[,Node...] | NodeFile | NodeClass}] [-o InodeResultFile]
                    [-P PoolName] [--inode-criteria CriteriaFile] [--qos QOSClass]
```

or

```
mmrestripefs Device {-r | -b | -R | -c [--read-only]} --metadata-only
                    [-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.

-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.

--metadata-only
Limits the specified operation to metadata blocks. Data blocks are not affected. This option is valid only with the **-r**, **-b**, **-R**, or**-c** option.

mmrestripefs command with this option completes its operation quicker than a full restripe, replication, or replica compare of data and metadata.

Use this option when you want to priortize the mmrestripefs operation on the metadata. This option ensures that the mmrestripefs operation has a reduced impact on the file system performance when compared to running the **mmrestripefs** command on the metadata and data.

After running the **mmrestripefs** command on the metadata with *--metadata-only* option, you can run the **mmrestripefs** command without this option to restripe the data and any metadata that requires to be restriped.

Note: This option does not run until all the nodes in the cluster are upgraded to IBM Spectrum Scale 4.2.1 release. If any of the nodes is not upgraded, the system displays the following error message:

```
mmrestripefs: The --metadata-only option support has not been enabled yet.
Run "mmchconfig release=LATEST" to activate the new function.
mmrestripefs: Command failed. Examine previous error messages to determine cause.
```

-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.

-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 Guide*.

-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 Guide*.

-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
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.

illCompressed
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.

illPlaced
Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

illReplicated
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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 --inode-criteria /tmp/crit -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

- "mmadddisk command" on page 23
- "mmapplypolicy command" on page 56
- "mmchattr command" on page 113
- "mmchdisk command" on page 151
- "mmchfs command" on page 169
- "mmdeldisk command" on page 266
- "mmrpldisk command" on page 485
- "mmrestripefile command" on page 475

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 **mmlsnsd -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 **mmlsnsd -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 **mmlsnsd -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: 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.

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 **mmadddisk** command to add the new disk, the **mmdeldisk** 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 Guide*.

--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.

illCompressed
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.

illPlaced
Indicates an inode with some data blocks that might be stored in an incorrect storage pool.

illReplicated
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
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.

--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" on page 196. 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: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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.
```

2. 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 ...
```

```
   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 23
- "mmchdisk command" on page 151
- "mmcrnsd command" on page 241
- "mmlsdisk command" on page 361
- "mmlsnsd command" on page 381
- "mmrestripefs command" on page 478

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 is 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 762.

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 Guide*. This command does not support a *NodeClass* of mount.

Exit status

0 Successful completion.

nonzero
: A failure occurred.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 762

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* in *IBM Spectrum Scale: Administration Guide*

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.

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 999999999999999K. 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 the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

Examples

1. 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
   Filesystem Fileset    type            KB      quota      limit   in_doubt    grace |    files   quota    limit in_doubt    grace  Remarks
   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
   ```

2. If perfileset 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:

```
# mmlsquota -g fvt090 fs1

Disk quotas for group fvt090 (gid 2590):
                         Block Limits                                    |     File Limits
Filesystem Fileset    type             KB      quota      limit   in_doubt    grace |    files   quota    limit in_doubt    grace  Remarks
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 159
- “mmdefedquota command” on page 251
- “mmdefquotaoff command” on page 254
- “mmdefquotaon command” on page 257
- “mmedquota command” on page 298
- “mmlsquota command” on page 391
- “mmquotaon command” on page 455
- “mmquotaoff command” on page 453
- “mmrepquota command” on page 463

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 Guide*.

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 × *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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
   c6f1c3vp2.gpfs.net:  forced unmount of /gpfs/fs1
   c6f1c3vp1.gpfs.net:  forced unmount of /gpfs/fs1
   Thu Mar 15 14:03:00 EDT 2012: mmshutdown: Shutting down GPFS daemons
   c6f1c3vp3.gpfs.net:  Shutting down!
   c6f1c3vp4.gpfs.net:  Shutting down!
   c6f1c3vp4.gpfs.net:  'shutdown' command about to kill process 23649
   c6f1c3vp4.gpfs.net:  Unloading modules from /lib/modules/2.6.27.19-5-ppc64/extra
   c6f1c3vp4.gpfs.net:  Unloading module mmfs26
   c6f1c3vp1.gpfs.net:  Shutting down!
   c6f1c3vp4.gpfs.net:  Unloading module mmfslinux
   c6f1c3vp1.gpfs.net:  'shutdown' command about to kill process 7667944
   c6f1c3vp2.gpfs.net:  Shutting down!
   c6f1c3vp2.gpfs.net:  'shutdown' command about to kill process 5701764
   c6f1c3vp2.gpfs.net:  Master did not clean up; attempting cleanup now
   c6f1c3vp2.gpfs.net:  Thu Mar 15 14:04:05.114 2012: mmfsd is shutting down.
   c6f1c3vp2.gpfs.net:  Thu Mar 15 14:04:05.115 2012: Reason for shutdown: mmfsadm shutdown command timed out
   c6f1c3vp2.gpfs.net:  Thu Mar 15 14:04:06 EDT 2012: mmcommon mmfsdown invoked.  Subsystem: mmfs Status: down
   c6f1c3vp2.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 320
- "mmlscluster command" on page 357
- "mmstartup command" on page 515

Location

```
/usr/lpp/mmfs/bin
```

mmsmb command

Administers SMB shares, export ACLs, and global configuration.

Synopsis

```
mmsmb export list [ListofSMBExports ][ -Y ][ --option Arg ][ --export-regex Arg ]
[ --header N ][ --all ][ --key-info Arg]
```

or

```
mmsmb export add SMBExport Path [--option SMBOption=Value | --key-info SMBOption]
```

or

```
mmsmb export change SMBExport [--option SMBOption=Value | --remove SMBOption | --key-info SMBOption]
```

or

```
mmsmb export remove SMBExport [--force]
```

or

```
mmsmb config list [ListofSMBOptions | -Y | --supported | --header N | --key-info SMBOption]
```

or

```
mmsmb config change [--option SMBOption=Value | --remove SMBOption | --key-info SMBOption]
```

or

```
mmsmb exportacl getid { Name | --user UserName | --group GroupName
| --system SystemName }
```

or

```
mmsmb exportacl list { [ExportName] | [ExportName --viewsddl] }
```

or

```
mmsmb exportacl add ExportName { Name | --user UserName | --group GroupName
| --system SystemName | --SID SID } --access Access --permissions Permissions [--force]
```

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 [--force]
```

or

```
mmsmb exportacl delete ExportName  [--force]
```

Note: For **mmsmb export**, you can specify −−option −−remove multiple times but you cannot specify both of these options simultaneously.

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 shares and global configuration.

Using the **mmsmb export** command, you can do the following tasks:

- Create the specified SMB share. 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 share using the **mmsmb export change** command.
- Delete an SMB share using the **mmsmb export remove** command. Existing connections to the deleted SMB share will be disconnected. This can result in data loss for files being currently open on the affected connections.
- List the SMB shares by using the **mmsmb export list** command. The command displays the configuration of options for each SMB share. If no specific options are specified, the command displays all SMB shares with the SMB options `browseable`; `guest ok`; `smb encrypt` as a table. Each row represents an SMB share 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 shares. Use the **mmsmb config list** command to display the global configuration options of SMB shares. 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 the SMB shares.

ListofSMBExports
Specifies the list of SMB shares 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 shares with unsupported SMB options. The path of all unsupported options are listed for each SMB share.

--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 share on a GPFS file system with NFSv4 ACLs enforced. You can verify whether your GPFS files system has been configured correctly by using the **mmlsfs** command. For example, `mmlsfs gpfs0 -k`.

path
Specifies the path of the SMB share 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 share.

--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 share.

--force
Suppresses confirmation questions.

SMBExport
Specifies the SMB share 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.

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 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:onclose
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 shares.

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 shares.

--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 myUser
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:

- {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 DENIED --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
mmsmb exportacl replace myUser --access ALLOWED --permissions FULL
```

delete

The Delete command will remove an entire 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

-Y 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 a "**%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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:dmapi support:ea support:fileid%3Amapping:force unknown
acl user:gencache%3Astabilize_count:gpfs%3Adfreequota:gpfs%3Ahsm: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:

```
Supported smb options with allowed values:
gpfs:dfreequota    = yes, no
restrict anonymous = 0, 2
server string      = any
```

Note: The output of this command depends on the configuration options supported by the system.

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 myExport --option all
   ```

   The system displays output similar to this:

   ```
   export          path                smb          encrypt          guest ok          browseable
   myExport        /gpfs/fs0/combo-1   auto         no               yes
   ```

2. To list SMB option csc policy for SMB share myexport and all SMB shares 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/testSmbEncypt   disable
   foo111b23   /mnt/gpfs0/testExport/testSmbEncypt   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 share 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 share 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 share `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 share `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 share `myExport` without confirmation:

```
mmsmb export remove myExport --force
```

The system removes the SMB share without any confirmation.

See also

- "mmnfs command" on page 402
- "mmces command" on page 96
- "mmlsfs command" on page 369

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 file systems* in the *IBM Spectrum Scale: Administration Guide*.

For more information on fileset snapshots, see *Fileset-level snapshots* in the *IBM Spectrum Scale: 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.

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 the topic *Requirements for administering a GPFS file system* in the *IBM Spectrum Scale: Administration Guide*.

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 for "fs1" is ".link" in root directory only
```

See also

- "mmcrsnapshot command" on page 246
- "mmdelsnapshot command" on page 283
- "mmlssnapshot command" on page 395
- "mmrestorefs command" on page 471

Location

```
/usr/lpp/mmfs/bin
```

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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 320
- "mmlscluster command" on page 357
- "mmshutdown command" on page 498

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.

--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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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:

```
mmlsconfig|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 pgalloc 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 123

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 files 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 Guide*.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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

- "mmmount command" on page 400
- "mmlsmount command" on page 377

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 IBM Spectrum Protect Backup-Archive client, use caution when you unlink filesets that contain data backed up by IBM Spectrum Protect. IBM Spectrum Protect tracks files by pathname and does not track filesets. As a result, when you unlink a fileset, it appears to IBM Spectrum Protect that you deleted the contents of the fileset. Therefore, the IBM Spectrum Protect Backup-Archive client inactivates the data on the IBM Spectrum Protect server which may result in the loss of backup data during the expiration process.

For information on GPFS filesets, see the *IBM Spectrum Scale: 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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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 163
- “mmcrfileset command” on page 223
- “mmdelfileset command” on page 271
- “mmlinkfileset command” on page 350
- “mmlsfileset command” on page 365

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 ldapserver.mydomain.com:1389`. Multiple LDAP servers can be specified is 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 ldapserver.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 ldapserver.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`.

--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).

--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.

--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-attrib or user-name-attrib 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 *externaleEndpoint*
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.

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 GPFS file system* in *IBM Spectrum Scale: Administration Guide*.

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
```

3. 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
-------------------------------------------------
```

4. 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
```

```
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
-------------------------------------------------
```

7. 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
```

```
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-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        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
-------------------------------------------------
```

```
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
```

```
 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 is 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 errorneous config files and service-related errors only.

See also

- “mmces command” on page 96
- “mmchconfig command” on page 123
- “mmlscluster command” on page 357
- “mmlsconfig command” on page 359
- “mmnfs command” on page 402
- “mmobj command” on page 414
- “mmsmb command” on page 500

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 *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.

--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 Guide*.

-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 [-l] [-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 perfmon [-l] [-r {on | off}]
```

or

```
spectrumscale config ntp [-e {on | off} [-l List ][-s Upstream_Servers]]
```

or

```
spectrumscale config clear {gpfs | protocols | object}
```

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**]

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 servers

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 auth commitsettings

or

spectrumscale auth clear

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] [-s] [-f]
```

or

```
spectrumscale deploy [-pr] [-po] [-s] [-f]
```

or

```
spectrumscale upgrade [-pr  | -po | -ve] [-f]
```

Availability

The **spectrumscale** is available as follows:

- Available with IBM Spectrum Scale Standard Edition or higher.

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 one of the following packages:
  - python-2.6 or later 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.

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:

-l Lists the current settings in the 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* in *IBM Spectrum Scale: Administration Guide*.

protocols
Provides details of the GPFS environment that will be used during protocol deployment, according to the following options:

-l Lists the current settings in the configuration.

-f *FileSystem*
Specifies the file system.

-m *MountPoint*
Specifies the mount point.

-e *ExportIPPool*
Specifies a comma-separated list of additional CES export IPs to configure on the cluster.

object
Sets any of the following Object-specific properties to be used during Object deployment and configuration:

-l Lists the current settings in the 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.

-ap *AdminPassword*
Specifies the admin user password. This credential is for the Keystone administrator. This user can be local or on remote authentication server based on the authentication type used.

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. This credential is for the Swift services administrator. All Swift services are run in this user's context. This user can be local or on remote authentication server based on the authentication type used.

-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 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.

-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.

perfmon
Sets Performance Monitoring specific properties to be used during installation and configuration:

-l Lists the current settings in the configuration.

-r on | off
Specifies if the installation toolkit can reconfigure Performance Monitoring.

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*
Lists the current settings of 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.

clear
Removes specified properties from the cluster definition file:

gpfs
Removes GPFS related properties from the cluster definition file:

-c Clears the GPFS cluster name.

-p Clears the GPFS profile to be applied on cluster creation. The following values are accepted:

default
Specifies that the **GpfsProtocolDefaults** profile is to be cleared.

randomio
Specifies that the **GpfsProtocolRandomIO** profile is to be cleared.

-r *RemoteShell*
Clears the absolute path name of the remote shell command GPFS uses for node communication. For example, `/usr/bin/ssh`.

-rc *RemoteFileCopy*
Clears the absolute path name of the remote copy command GPFS uses when transferring files between nodes. For example, `/usr/bin/scp`.

-e *EphemeralPortRange*
Clears the GPFS daemon communication port range.

--all
Clears all settings in the cluster definition file.

protocols
Removes protocols related properties from the cluster definition file:

-f Clears the shared file system name.

-m Clears the shared file system mount point.

-e Clears a comma-separated list of additional CES export IPs to configure on the cluster.

--all
Clears all settings in the cluster definition file.

object
Removes object related properties from the cluster definition file:

-f Clears the object file system name.

-m Clears the absolute path to your file system on which the objects reside.

-e Clears the host name which maps to all CES IP addresses in a round-robin manner.

-o Clears the GPFS fileset to be created or used as the object base.

-i Clears the GPFS fileset inode allocation to be used by the object base.

-t Clears the admin token to be used by Keystone.

-au
Clears the user name for the admin user.

-ap
Clears the password for the admin user.

-su
Clears the user name for the Swift user.

-sp
Clears the password for the Swift user.

-dp
Clears the password for the object database.

-s3
Clears the S3 API setting, if it is enabled.

-mr
Clears the multi-region data file path.

-rn
Clears the region number for the multi-region configuration.

--all
Clears all settings in the cluster definition file.

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

-fs *FileSystem*
Specifies the file system.

-fg *FailureGroup*
Specifies the failure group.

servers
Adds and removes servers, and sets the primary server for NSDs.

clear
Clears the current NSD configuration. The following option is accepted:

-f Forces the action without manual confirmation.

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.

commitsettings
Merges authentication settings into the main cluster definition file.

clear
Clears your current authentication configuration.

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.

-po
Performs a post-install environment check.

-s *SecretKey*
Specifies the secret key on the command line required to decrypt sensitive data in the cluster definition file and suppresses the prompt for the secret key.

-f Forces action without manual confirmation.

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.

-s *SecretKey*
Specifies the secret key on the command line required to decrypt sensitive data in the cluster definition file and suppresses the prompt for the secret key.

-f Forces action without manual confirmation.

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

-f Forces action without manual confirmation.

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. For more information, see *Requirements for administering a GPFS(tm) file system* in *IBM Spectrum Scale: Administration Guide*.

Examples

Creating a new IBM Spectrum Scale cluster

1. To instantiate your chef zero server, issue a command similar to the following:
   ```
   spectrumscale setup -s 192.168.0.1
   ```
2. To designate NSD server nodes in your environment to use for the installation, issue this command:
   ```
   ./spectrumscale node add FQDN -n
   ```
3. To add four non-shared NSDs seen by a primary NSD server only, issue this command:
   ```
   ./spectrumscale nsd add -p FQDN_of_Primary_NSD_Server /dev/dm-1 /dev/dm-2 /dev/dm-3 /dev/dm-4
   ```
4. To add four non-shared NSDs seen by both a primary NSD server and a secondary NSD server, issue this command:
   ```
   ./spectrumscale nsd add -p FQDN_of_Primary_NSD_Server -s FQDN_of_Secondary_NSD_Server\
    /dev/dm-1 /dev/dm-2 /dev/dm-3 /dev/dm-4
   ```

5. To define a shared root file system using two NSDs and a file system `fs1` using two NSDs, issue these commands:

   ```
   ./spectrumscale nsd list
   ./spectrumscale file system list
   ./spectrumscale nsd modify nsd1 -fs cesSharedRoot
   ./spectrumscale nsd modify nsd2 -fs cesSharedRoot
   ./spectrumscale nsd modify nsd3 -fs fs1
   ./spectrumscale nsd modify nsd4 -fs fs1
   ```

6. To designate GUI nodes in your environment to use for the installation, issue this command:

   ```
   ./spectrumscale node add FQDN -g -a
   ```

7. To designate additional client nodes in your environment to use for the installation, issue this command:

   ```
   ./spectrumscale node add FQDN
   ```

8. To allow the installation toolkit to reconfigure Performance Monitoring if it detects any existing configurations, issue this command:

   ```
   ./spectrumscale config perfmon -r on
   ```

9. To name your cluster, issue this command:

   ```
   ./spectrumscale config config gpfs -c Cluster_Name
   ```

10. To review the configuration prior to installation, issue these commands:

    ```
    ./spectrumscale node list
    ./spectrumscale nsd list
    ./spectrumscale filesystem list
    ./spectrumscale config gpfs --list
    ```

11. To start the installation on your defined environment, issue these commands:

    ```
    ./spectrumscale install --precheck
    ./spectrumscale install
    ```

12. To deploy file systems after a successful installation, do one of the following depending on your requirement:
    - If you want to use only the file systems, issue these commands:

      ```
      ./spectrumscale deploy --precheck
      ./spectrumscale deploy
      ```

    - If you want to deploy protocols also, see the examples in the *Deploying protocols on an existing cluster* section.

Deploying protocols on an existing cluster

Note: If your cluster contains ESS, see the *Adding protocols to a cluster containing ESS* section.

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 deployment, issue this command:

   ```
   ./spectrumscale node add FQDN -p
   ```

3. To designate GUI nodes in your environment to use for the deployment, issue this command:

   ```
   ./spectrumscale node add FQDN -g -a
   ```

4. To configure protocols to point to a file system that will be used as a shared root, issue this command:

   ```
   ./spectrumscale config protocols -f FS_Name -m FS_Mountpoint
   ```

5. To configure a pool of export IPs, issue this command:

   ```
   ./spectrumscale config protocols -e Comma_Separated_List_of_Exportpool_IPs
   ```

6. To enable NFS on all protocol nodes, issue this command:

   ```
   ./spectrumscale enable nfs
   ```

7. To enable SMB on all protocol nodes, issue this command:

```
./spectrumscale enable smb
```

8. To enable Object on all protocol nodes, issue these commands:

```
./spectrumscale enable object
./spectrumscale config object -au Admin_User -ap Admin_Password -dp Database_Password
./spectrumscale config object -e FQDN
./spectrumscale config object -f FS_Name -m FS_Mountpoint
./spectrumscale config object -o Object_Fileset
```

9. To review the configuration prior to deployment, issue these commands:

```
./spectrumscale config protocols
./spectrumscale config object
./spectrumscale node list
```

10. To deploy protocols on your defined environment, issue these commands:

```
./spectrumscale deploy --precheck
./spectrumscale deploy
```

Deploying protocol authentication

Note: For the following example commands, it is assumed that the protocols cluster was deployed successfully using the **spectrumscale** command options.

1. To enable file authentication with AD server on all protocol nodes, issue this command:

   ```
   ./spectrumscale auth file ad
   ```

   Fill out the template and save the information, and then issue the following commands:

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

2. To enable Object authentication with AD server on all protocol nodes, issue this command:

   ```
   ./spectrumscale auth object ad
   ```

   Fill out the template and save the information, and then issue the following commands:

   ```
   ./spectrumscale deploy --precheck
   ./spectrumscale deploy
   ```

Upgrading an IBM Spectrum Scale cluster

1. Extract the IBM Spectrum Scale package for the required code level by issuing a command similar to the following depending on the package name:

   ```
   ./Spectrum_Scale_Protocols_Standard-4.2.x.x-xxxxx
   ```

2. Copy the cluster definition file file from the prior installation to the latest installer location by issuing this command:

   ```
   cp -p /usr/lpp/mmfs/4.2.0.1/installer/configuration/clusterdefinition.txt\
    /usr/lpp/mmfs/4.2.1.0/installer/configuration/
   ```

   Note: This is a command example of when you are upgrading from 4.2.0.1 to 4.2.1.0.

3. Run the upgrade precheck from the installer directory of the latest code level extraction by issuing commands similar to the following:

   ```
   cd /usr/lpp/mmfs/Latest_Code_Level_Directory/installer
   ./spectrumscale upgrade --precheck
   ```

4. Run the upgrade by issuing this command:

   ```
   cd /usr/lpp/mmfs/Latest_Code_Level_Directory/installer
   ./spectrumscale upgrade
   ```

Adding to an installation process

1. To add nodes to an installation, do the following:
   a. Add one or more node types using the following commands:
      - Client nodes:

        ```
        ./spectrumscale node add FQDN
        ```

- NSD nodes:

  ```
  ./spectrumscale node add FQDN -n
  ```

- Protocol nodes:

  ```
  ./spectrumscale node add FQDN -p
  ```

- GUI nodes:

  ```
  ./spectrumscale node add FQDN -g -a
  ```

b. Install GPFS on the new nodes using the following commands:

```
./spectrumscale install --precheck
./spectrumscale install
```

c. If protocol nodes are being added, deploy protocols using the following commands:

```
./spectrumscale deploy --precheck
./spectrumscale deploy
```

2. To add NSDs to an installation, do the following:
   a. Verify that the NSD server connecting this new disk runs an OS compatible with the **spectrumscale** installation toolkit and that the NSD server exists within the cluster.
   b. Add NSDs to the installation using the following command:

      ```
      ./spectrumscale nsd add -p FQDN_of_Primary_NSD_Server Path_to_Disk_Device_File
      ```

   c. Run the installation using the following commands:

      ```
      ./spectrumscale install --precheck
      ./spectrumscale install
      ```

3. To add file systems to an installation, do the following:
   a. Verify that free NSDs exist and that they can be listed by the **spectrumscale** installation toolkit using the following commands.

      ```
      mmlsnsd
      ./spectrumscale nsd list
      ```

   b. Define the file system using the following command:

      ```
      ./spectrumscale nsd modify NSD -fs File_System_Name
      ```

   c. Deploy the file system using the following commands:

      ```
      ./spectrumscale deploy --precheck
      ./spectrumscale deploy
      ```

4. To enable another protocol on an existing cluster that has protocols enabled, do the following steps depending on your configuration:
   a. Enable NFS on all protocol nodes using the following command:

      ```
      ./spectrumscale enable nfs
      ```

   b. Enable SMB on all protocol nodes using the following command:

      ```
      ./spectrumscale enable smb
      ```

   c. Enable Object on all protocol nodes using the following commands:

      ```
      ./spectrumscale enable object
      ./spectrumscale config object -au Admin_User -ap Admin_Password -dp Database_Password
      ./spectrumscale config object -e FQDN
      ./spectrumscale config object -f FS_Name -m FS_Mountpoint
      ./spectrumscale config object -o Object_Fileset
      ```

   d. Enable the new protocol using the following commands:

      ```
      ./spectrumscale deploy --precheck
      ./spectrumscale deploy
      ```

Adding protocols to a cluster containing ESS

For information on preparing a cluster that contains ESS for adding protocols, see *Preparing a cluster that contains ESS for adding protocols* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

After you have prepared your cluster that contains ESS for adding protocols, you can use commands similar to the ones listed in the *Deploying protocols on an existing cluster* section.

Diagnosing an error during install, deploy, or upgrade

1. Note the screen output indicating the error. This helps in narrowing down the general failure.
   When a failure occurs, the screen output also shows the log file containing the failure.
2. Open the log file in an editor such as `vi`.
3. Go to the end of the log file and search upwards for the text `FATAL`.
4. Find the topmost occurrence of FATAL (or first FATAL error that occurred) and look above and below this error for further indications of the failure.

For more information, see *Finding deployment related error messages more easily and using them for failure analysis* in *IBM Spectrum Scale: Problem Determination Guide*.

See also

- *Configuring with the spectrumscale installation toolkit* in the *IBM Spectrum Scale: Administration Guide*.
- "mmchconfig command" on page 123
- "mmlscluster command" on page 357
- "mmlsconfig command" on page 359
- "mmnfs command" on page 402
- "mmobj command" on page 414
- "mmsmb command" on page 500
- "mmuserauth command" on page 527

See also the topic about installing IBM Spectrum Scale on Linux nodes in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Location

```
/usr/lpp/mmfs/4.2.1.0/installer
```

Chapter 2. IBM Spectrum Scale Data Management API for GPFS information

The Data Management Application Programming Interface (DMAPI) for (GPFS) is based on The Open Group's System Management: Data Storage Management (XDSM) 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.

The XDSM 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.

Overview of IBM Spectrum Scale Data Management API for GPFS

The Data Management Application Programming Interface (DMAPI) for GPFS allows you to monitor events associated with a GPFS file system or with an individual file. You can also manage and maintain file system data.

See the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) for the current limitations of DMAPI-managed file systems.

Note: IBM Spectrum Protect for Space Management for GPFS file systems is not available for Windows.

DMAPI for GPFS is compliant with the Open Group's XDSM Standard and includes these features:

- "GPFS-specific DMAPI events"
- "DMAPI functions" on page 566
- "DMAPI configuration attributes" on page 570
- "DMAPI restrictions for GPFS" on page 571

GPFS-specific DMAPI events

There are three GPFS-specific DMAPI events: events implemented in DMAPI for GPFS, optional events that are not implemented in DMAPI for GPFS, and GPFS-specific attribute events that are not part of the DMAPI standard.

For more information, see:

- "Events implemented in DMAPI for GPFS"
- "Optional events that are not implemented in DMAPI for GPFS" on page 566
- "GPFS-specific attribute events that are not part of the DMAPI standard" on page 566

Events implemented in DMAPI for GPFS

These are the events, as defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, implemented in DMAPI for GPFS:

File system administration events

- mount
- preunmount
- unmount
- nospace

© Copyright IBM Corporation © IBM 2014, 2016

Namespace events

- create, postcreate
- remove, postremove
- rename, postrename
- symlink, postsymlink
- link, postlink

Data events

- read
- write
- truncate

Metadata events

- attribute
- destroy
- close

Pseudo event

- user event

GPFS guarantees that asynchronous events are delivered, except when the GPFS daemon fails. Events are enqueued to the session before the corresponding file operation completes. For further information on failures, see "Failure and recovery of IBM Spectrum Scale Data Management API for GPFS" on page 601.

Optional events that are not implemented in DMAPI for GPFS

The following optional events, as defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, are **not** implemented in DMAPI for GPFS:

File system administration event

- debut

Metadata event

- cancel

GPFS-specific attribute events that are not part of the DMAPI standard

GPFS generates the following attribute events for DMAPI that are specific to GPFS and not part of the DMAPI standard:

- Pre-permission change
- Post-permission change

For additional information, refer to "GPFS-specific DMAPI events" on page 599.

DMAPI functions

All mandatory DMAPI functions and most optional functions that are defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, are implemented in DMAPI for GPFS.

For C declarations of all the functions implemented in DMAPI for GPFS, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory.

For changes and restrictions on functions in DMAPI for GPFS, see “Usage restrictions on DMAPI functions” on page 583, and “Semantic changes to DMAPI functions” on page 598.

Mandatory functions implemented in DMAPI for GPFS

These mandatory functions, as defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, are implemented in DMAPI for GPFS.

For C declarations of all the mandatory functions implemented in DMAPI for GPFS, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory. However, for a quick description of the mandatory functions and their applications, refer to the following set of functions:

dm_create_session
: Create a new session.

dm_create_userevent
: Create a pseudo-event message for a user.

dm_destroy_session
: Destroy an existing session.

dm_fd_to_handle
: Create a file handle using a file descriptor.

dm_find_eventmsg
: Return the message for an event.

dm_get_allocinfo
: Get a file's current allocation information.

dm_get_bulkattr
: Get bulk attributes of a file system.

dm_get_config
: Get specific data on DMAPI implementation.

dm_get_config_events
: List all events supported by the DMAPI implementation.

dm_get_dirattrs
: Return a directory's bulk attributes.

dm_get_eventlist
: Return a list of an object's enabled events.

dm_get_events
: Return the next available event messages.

dm_get_fileattr
: Get file attributes.

dm_get_mountinfo
: Return details from a mount event.

dm_get_region
: Get a file's managed regions.

dm_getall_disp
: For a given session, return the disposition of all file system's events.

dm_getall_sessions
: Return all extant sessions.

dm_getall_tokens
: Return a session's outstanding tokens.

dm_handle_cmp
Compare file handles.

dm_handle_free
Free a handle's storage.

dm_handle_hash
Hash the contents of a handle.

dm_handle_is_valid
Check a handle's validity.

dm_handle_to_fshandle
Return the file system handle associated with an object handle.

dm_handle_to_path
Return a path name from a file system handle.

dm_init_attrloc
Initialize a bulk attribute location offset.

dm_init_service
Initialization processing that is implementation-specific.

dm_move_event
Move an event from one session to another.

dm_path_to_fshandle
Create a file system handle using a path name.

dm_path_to_handle
Create a file handle using a path name.

dm_query_right
Determine an object's access rights.

dm_query_session
Query a session.

dm_read_invis
Read a file without using DMAPI events.

dm_release_right
Release an object's access rights.

dm_request_right
Request an object's access rights.

dm_respond_event
Issue a response to an event.

dm_send_msg
Send a message to a session.

dm_set_disp
For a given session, set the disposition of all file system's events.

dm_set_eventlist
For a given object, set the list of events to be enabled.

dm_set_fileattr
Set a file's time stamps, ownership and mode.

dm_set_region
Set a file's managed regions.

dm_write_invis
Write to a file without using DMAPI events.

Optional functions implemented in DMAPI for GPFS

These optional functions, as defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, are implemented in DMAPI for GPFS.

For C declarations of these optional functions implemented in DMAPI for GPFS, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory. However, for a quick description of the optional functions and their applications, refer to the following set of functions:

dm_downgrade_right
Change an exclusive access right to a shared access right.

dm_get_bulkall
Return a file system's bulk data management attributes.

dm_get_dmattr
Return a data management attribute.

dm_getall_dmattr
Return all data management attributes of a file.

dm_handle_to_fsid
Get a file system ID using its handle.

dm_handle_to_igen
Get inode generation count using a handle.

dm_handle_to_ino
Get inode from a handle.

dm_make_handle
Create a DMAPI object handle.

dm_make_fshandle
Create a DMAPI file system handle.

dm_punch_hole
Make a hole in a file.

dm_probe_hole
Calculate the rounded result of the area where it is assumed that a hole is to be punched.

dm_remove_dmattr
Delete a data management attribute.

dm_set_dmattr
Define or update a data management attribute.

dm_set_return_on_destroy
Indicate a DM attribute to return with destroy events.

dm_sync_by_handle
Synchronize the in-memory state of a file with the physical medium.

dm_upgrade_right
Change a currently held access right to be exclusive.

Optional functions that are not implemented in DMAPI for GPFS

There are optional functions that are not implemented in DMAPI for GPFS.

The following optional functions, as defined in the *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X, are **not** implemented in DMAPI for GPFS:

dm_clear_inherit
Reset the inherit-on-create status of an attribute.

dm_create_by_handle
Define a file system object using a DM handle.

dm_getall_inherit
Return a file system's inheritable attributes.

dm_mkdir_by_handle
Define a directory object using a handle.

dm_obj_ref_hold
Put a hold on a file system object.

dm_obj_ref_query
Determine if there is a hold on a file system object.

dm_obj_ref_rele
Release the hold on a file system object.

dm_pending
Notify FS of slow DM application processing.

dm_set_inherit
Indicate that an attribute is inheritable.

dm_symlink_by_handle
Define a symbolic link using a DM handle.

GPFS-specific DMAPI functions

There are several GPFS-specific DMAPI functions that are not part of the DMAPI open standard.

The GPFS-specific functions are listed and described in "Definitions for GPFS-specific DMAPI functions" on page 584.

DMAPI configuration attributes

The *System Management: Data Storage Management (XDSM) API* Common Applications Environment (CAE) Specification C429, The Open Group, ISBN 1-85912-190-X defines a set of configuration attributes to be exported by each DMAPI implementation. These attributes specify which optional features are supported and give bounds on various resources.

The Data Management (DM) application can query the attribute values using the function **dm_get_config**. It can also query which events are supported, using the function **dm_get_config_events**.

The functions **dm_get_config** and **dm_get_config_events** receive a file handle from input arguments *hanp* and *hlen*. In GPFS, both functions ignore the handle, as the configuration is not dependent on the specific file or file system. This enables the DM application to query the configuration during initialization, when file handles may not yet be available.

Note: To guarantee that the most current values are being used, the DM application should always query the configuration at runtime by using **dm_get_config**.

Table 16 on page 571 shows the attribute values that are used in DMAPI for GPFS:

Table 16. DMAPI configuration attributes

Name	Value
DM_CONFIG_BULKALL	1
DM_CONFIG_CREATE_BY_HANDLE	0
DM_CONFIG_DTIME_OVERLOAD	1
DM_CONFIG_LEGACY	1
DM_CONFIG_LOCK_UPGRADE	1
DM_CONFIG_MAX_ATTR_ON_DESTROY	1022
DM_CONFIG_MAX_ATTRIBUTE_SIZE	1022
DM_CONFIG_MAX_HANDLE_SIZE	32
DM_CONFIG_MAX_MANAGED_REGIONS	32
DM_CONFIG_MAX_MESSAGE_DATA	4096
DM_CONFIG_OBJ_REF	0
DM_CONFIG_PENDING	0
DM_CONFIG_PERS_ATTRIBUTE	1
DM_CONFIG_PERS_EVENTS	1
DM_CONFIG_PERS_INHERIT_ATTRIBS	0
DM_CONFIG_PERS_MANAGED_REGIONS	1
DM_CONFIG_PUNCH_HOLE	1
DM_CONFIG_TOTAL_ATTRIBUTE_SPACE	7168
DM_CONFIG_WILL_RETRY	0

Attribute value **DM_CONFIG_TOTAL_ATTRIBUTE_SPACE** is per file. The entire space is available for opaque attributes. Non-opaque attributes (event list and managed regions) use separate space.

DMAPI restrictions for GPFS

All DMAPI APIs must be called from nodes that are in the cluster where the file system is created. DMAPI APIs may **not** be invoked from a remote cluster.

Furthermore, GPFS also places the following DMAPI API restrictions:

- Running **dm_get_events** with the **DM_EV_WAIT** flag set causes the calling process to wait uninterruptibly.
- Interacting with a handle after calling **dm_handle_free** will result in undefined behavior.

In addition to the DMAPI API restrictions, GPFS places the following restrictions on the use of file system snapshots when you have DMAPI enabled:

- Snapshots cannot coexist with file systems using GPFS 3.1 or earlier.
- GPFS 3.2 and later permits snapshots with DMAPI-enabled file systems. However, GPFS places the following restrictions on DMAPI access to the snapshot files:
  - The DM server may read files in a snapshot using **dm_read_invis**.
  - The DM server is not allowed to modify or delete the file using **dm_write_invis** or **dm_punch_hole**.
  - The DM server is not allowed to establish a managed region on the file.
  - Snapshot creation or deletion does not generate DMAPI namespace events.
  - Snapshots of a file are not managed regardless of the state of the original file and they do not inherit the DMAPI attributes of the original file.

Concepts of IBM Spectrum Scale Data Management API for GPFS

The XDSM standard is intended mainly for a single-node environment. Some of the key concepts in the standard such as sessions, event delivery, mount and unmount, and failure and recovery, are not well defined for a multiple-node environment such as GPFS.

For a list of restrictions and coexistence considerations, see "Usage restrictions on DMAPI functions" on page 583.

All DMAPI APIs must be called from nodes that are in the cluster where the file system is created.

Key concepts of DMAPI for GPFS include these areas:

- "Sessions"
- "Data management events"
- "Mount and unmount" on page 574
- "Tokens and access rights" on page 575
- "Parallelism in Data Management applications" on page 576
- "Data Management attributes" on page 577
- "Support for NFS" on page 577
- "Quota" on page 577
- "Memory mapped files" on page 577

Sessions

In GPFS, a session is associated only with the node on which the session was created. This node is known as the *session node*.

Events are generated at any node where the file system is mounted. The node on which a given event is generated is called the *source node* of that event. The event is delivered to a session queue on the session node.

There are restrictions as to which DMAPI functions can and cannot be called from a node other than the session node. In general, functions that change the state of a session or event can only be called on the session node. For example, the maximum number of DMAPI sessions that can be created on a node is 4000. See "Usage restrictions on DMAPI functions" on page 583 for details.

Session ids are unique over time within a GPFS cluster. When an existing session is assumed, using **dm_create_session**, the new session id returned is the same as the old session id.

A session fails when the GPFS daemon fails on the session node. Unless this is a total failure of GPFS on all nodes, the session is recoverable. The DM application is expected to assume the old session, possibly on another node. This will trigger the reconstruction of the session queue. All pending synchronous events from surviving nodes are resubmitted to the recovered session queue. Such events will have the same token id as before the failure, except mount events. Asynchronous events, on the other hand, are lost when the session fails. See "Failure and recovery of IBM Spectrum Scale Data Management API for GPFS" on page 601 for information on failure and recovery.

Data management events

Data management events arrive on a session queue from any of the nodes in the GPFS cluster.

The source node of the event is identified by the **ev_nodeid** field in the header of each event message in the structure **dm_eventmsg**. The identification is the GPFS cluster data node number, which is attribute **node_number** in the **mmsdrfs2** file for a PSSP node or **mmsdrfs** file for any other type of node.

Data Management events are generated only if the following two conditions are true:

1. The event is enabled.
2. It has a disposition.

A file operation will fail with the **EIO** error if there is no disposition for an event that is enabled and would otherwise be generated.

A list of enabled events can be associated individually with a file and globally with an entire file system. The XDSM standard leaves undefined the situation where the individual and the global event lists are in conflict. In GPFS, such conflicts are resolved by always using the individual event list, if it exists.

Note: The XDSM standard does not provide the means to remove the individual event list of a file. Thus, there is no way to enable or disable an event for an entire file system without explicitly changing each conflicting individual event list.

In GPFS, event lists are persistent.

Event dispositions are specified per file system and are not persistent. They must be set explicitly after the session is created.

Event generation mechanisms have limited capacity. In case resources are exceeded, new file operations will wait indefinitely for free resources.

File operations wait indefinitely for a response from synchronous events. The **dmapiEventTimeout** configuration attribute on the **mmchconfig** command, can be used to set a timeout on events that originate from NFS file operations. This is necessary because NFS servers have a limited number of threads that cannot be blocked for long periods of time. Refer to "GPFS configuration attributes for DMAPI" on page 579 and "Support for NFS" on page 577.

The XDSM standard permits asynchronous events to be discarded at any time. In GPFS, asynchronous events are guaranteed when the system runs normally, but may be lost during abnormal conditions, such as failure of GPFS on the session node. Asynchronous events are delivered in a timely manner. That is, an asynchronous event is enqueued to the session before the corresponding file operation completes.

Figure 1 on page 574, shows the flow of a typical synchronous event in a multiple-node GPFS environment. The numbered arrows in the figure correspond to the following steps:

1. The user application on the source node performs a file operation on a GPFS file. The file operation thread generates a synchronous event and blocks, waiting for a response.
2. GPFS on the source node sends the event to GPFS on the session node, according to the disposition for that event. The event is enqueued to the session queue on the session node.
3. The Data Management application on the session node receives the event (using **dm_get_events**) and handles it.
4. The Data Management application on the session node responds to the event (using **dm_respond_event**).
5. GPFS on the session node sends the response to GPFS on the source node.
6. GPFS on the source node passes the response to the file operation thread and unblocks it. The file operation continues.

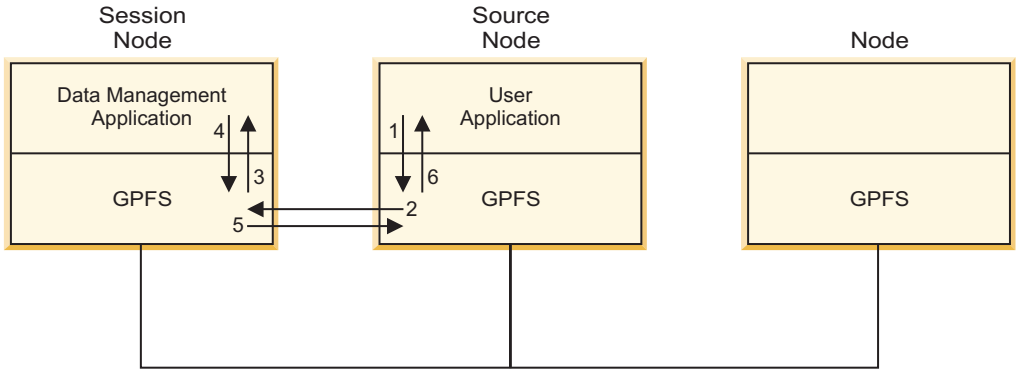

Figure 1. Flow of a typical synchronous event in a multiple-node GPFS environment

Reliable DMAPI destroy events

A metadata destroy event is generated when the operating system has destroyed an object. This type of event is different from a remove event, which is a namespace event and is not related to the destruction of an object. A reliable destroy event supports synchronous destroy events in the same way that other synchronous events do. When a synchronous event is generated, a user process is suspended in the kernel; it will be suspended until a DM application issues an explicit response to the event. The DM application at the session that supports the reliable destroy event must be capable of handling the synchronous destroy event. In other words, it must respond to the **DM_EVENT_DESTROY** event with **DM_RESPOND_EVENT**. Otherwise, the event will wait forever at the session node for a response. Based on this, it is recommended that the cluster not be made up of nodes that are running back-level code and new code, because the destroy event is not reliable in a mixed environment.

Mount and unmount

The XDSM standard implicitly assumes that there is a single mount, pre-unmount and unmount event per file system. In GPFS, a separate mount event is generated by each mount operation on each node. Similarly, if the pre-unmount and unmount events are enabled, they are generated by each unmount operation on each node. Thus, there may be multiple such events for the same file system.

To provide additional information to the DM application, the mode field in the respective event message structures (**me_mode** for mount, and **ne_mode** for pre-unmount and unmount) has a new flag, **DM_LOCAL_MOUNT**, which is not defined in the standard. When the flag is set, the mount or unmount operation is local to the session node. In addition, the new field **ev_nodeid** in the header of the event message can be used to identify the source node where the mount or unmount operation was invoked. The identification is the GPFS cluster data node number, which is attribute **node_number** in the **mmsdrfs2** file for a PSSP node or **mmsdrfs** file for any other type of node.

The mount event is sent to multiple sessions that have a disposition for it. If there is no disposition for the mount event, the mount operation fails with an **EIO** error.

There is no practical way to designate the *last* unmount, since there is no serialization of all mount and unmount operations of each file system. Receiving an unmount event with the value 0 in the **ne_retcode** field is no indication that there will be no further events from the file system.

An unmount initiated internally by the GPFS daemon, due to file system forced unmount or daemon shutdown, will not generate any events. Consequently, there need not be a match between the number of mount events and the number of pre-unmount or unmount events for a given file system.

The **dmapiMountTimeout** attribute on the **mmchconfig** command enables blocking the mount operation for a limited time until some session has set the mount disposition. This helps GPFS and the DM application synchronize during initialization. See "GPFS configuration attributes for DMAPI" on page 579 and "Initializing the Data Management application" on page 581.

Mount events are enqueued on the session queue ahead of any other events. This gives mount events a higher priority, which improves the response time for mount events when the queue is very busy.

If the **DM_UNMOUNT_FORCE** flag is set in the pre-unmount event message, the response of the DM application to the pre-unmount event is ignored, and the forced unmount proceeds. If the **DM_LOCAL_MOUNT** flag is also set, the forced unmount will result in the loss of all access rights of the given file system that are associated with any local session.

If the unmount is not forced (the **DM_UNMOUNT_FORCE** flag is not set), and the **DM_LOCAL_MOUNT** flag is set, the DM application is expected to release all access rights on files of the given file system associated with any local session. If any access rights remain held after the **DM_RESP_CONTINUE** response is given, the unmount will fail with **EBUSY**. This is because access rights render the file system busy, similar to other locks on files.

The function **dm_get_mountinfo** can be called from any node, even if the file system is not mounted on that node. The **dm_mount_event** structure returned by the **dm_get_mountinfo** function provides the following enhanced information. The **me_mode** field contains two new flags, **DM_LOCAL_MOUNT** and **DM_REMOTE_MOUNT**. At least one of the two flags is always set. When both flags are set simultaneously, it is an indication that the file system is mounted on the local node, as well as one or more other (remote) nodes. When only **DM_LOCAL_MOUNT** is set, it is an indication that the file system is mounted on the local node but not on any other node. When only **DM_REMOTE_MOUNT** is set, it is an indication that the file system is mounted on some remote node, but not on the local node.

In the latter case (only **DM_REMOTE_MOUNT** is set), the fields **me_roothandle** and **me_handle2** (the mount point handle) in the **dm_mount_event** structure are set to **DM_INVALID_HANDLE**. Also in this case, the **me_name1** field (the mount point path) is taken from the stanza in the file **/etc/filesystems** on one of the remote nodes (with the use of GPFS cluster data, the stanzas on all nodes are identical).

The enhanced information provided by the **dm_get_mountinfo** function can be useful during the processing of mount and pre-unmount events. For example, before responding to a mount event from a remote (non-session) node, **dm_get_mountinfo** could be invoked to find out whether the file system is already mounted locally at the session node, and if not, initiate a local mount. On receiving a pre-unmount event from the local session node, it is possible to find out whether the file system is still mounted elsewhere, and if so, fail the local unmount or delay the response until after all remote nodes have unmounted the file system.

Note: The **DM_REMOTE_MOUNT** flag is redundant in the **dm_mount_event** structure obtained from the mount event (as opposed to the **dm_get_mountinfo** function).

Tokens and access rights

A DMAPI token is an identifier of an outstanding event (a synchronous event that the DM application has received and is currently handling). The token is unique over time in the cluster. The token becomes invalid when the event receives a response.

The main purpose of tokens is to convey access rights in DMAPI functions. Access rights are associated with a specific event token. A function requiring access rights to some file may present an event token that has the proper access rights.

DMAPI functions can also be invoked using **DM_NO_TOKEN**, in which case sufficient access protection is provided for the duration of the operation. This is semantically equivalent to holding an access right, but no access right on the file is actually acquired.

In GPFS, when an event is received, its token has no associated access rights.

DM access rights are implemented in GPFS using an internal lock on the file. Access rights can be acquired, changed, queried, and released only at the session node. This is an implementation restriction caused by the GPFS locking mechanisms.

In GPFS, it is not possible to set an access right on an entire file system from the file system handle. Thus, DMAPI function calls that reference a file system, using a file system handle, are not allowed to present a token and must specify **DM_NO_TOKEN**. For the same reason, functions that acquire or change access rights are not allowed to present a file system handle.

Holding access rights renders the corresponding file system busy at the session node, preventing normal (non-forced) unmount. This behavior is similar to that of other locks on files. When receiving a pre-unmount event, the DM application is expected to release all access rights before responding. Otherwise, the unmount operation will fail with an **EBUSY** error.

All access rights associated with an event token are released when the response is given. There is no transfer of access rights from DMAPI to the file operation thread. The file operation will acquire any necessary locks after receiving the response of the event.

Parallelism in Data Management applications

Given the multiple-node environment of GPFS, it is desirable to exploit parallelism in the Data Management application as well.

This can be accomplished in several ways:

- On a given session node, multiple DM application threads can access the same file in parallel, using the same session. There is no limit on the number of threads that can invoke DMAPI functions simultaneously on each node.
- Multiple sessions, each with event dispositions for a different file system, can be created on separate nodes. Thus, files in different file systems can be accessed independently and simultaneously, from different session nodes.
- Dispositions for events of the same file system can be partitioned among multiple sessions, each on a different node. This distributes the management of one file system among several session nodes.
- Although GPFS routes all events to a single session node, data movement may occur on multiple nodes. The function calls **dm_read_invis**, **dm_write_invis**, **dm_probe_hole**, and **dm_punch_hole** are honored from a root process on another node, provided it presents a session ID for an established session on the session node.

  A DM application may create a *worker process*, which exists on any node within the GPFS cluster. This worker process can move data to or from GPFS using the **dm_read_invis** and **dm_write_invis** functions. The worker processes must adhere to these guidelines:

  1. They must run as root.
  2. They must present a valid session ID that was obtained on the session node.
  3. All writes to the same file which are done in parallel must be done in multiples of the file system block size, to allow correct management of disk blocks on the writes.
  4. No DMAPI calls other than **dm_read_invis**, **dm_write_invis**, **dm_probe_hole**, and **dm_punch_hole** may be issued on nodes other than the session node. This means that any rights required on a file must be obtained within the session on the session node, prior to the data movement.

5. There is no persistent state on the nodes hosting the worker process. It is the responsibility of the DM application to recover any failure which results from the failure of GPFS or the data movement process.

Data Management attributes

Data Management attributes can be associated with any individual file. There are opaque and non-opaque attributes.

An opaque attribute has a unique name, and a byte string value which is not interpreted by the DMAPI implementation. Non-opaque attributes, such as managed regions and event lists, are used internally by the DMAPI implementation.

DM attributes are persistent. They are kept in a hidden file in the file system.

GPFS provides two *quick access* single-bit opaque DM attributes for each file, stored directly in the inode. These attributes are accessible through regular DMAPI functions, by specifying the reserved attribute names **_GPFSQA1** and **_GPFSQA2** (where **_GPF** is a reserved prefix). The attribute data must be a single byte with contents 0 or 1.

Support for NFS

A DM application could be slow in handling events. NFS servers have a limited number of threads which must not all be blocked simultaneously for extended periods of time. GPFS provides a mechanism to guarantee progress of NFS file operations that generate data events without blocking the server threads indefinitely.

The mechanism uses a timeout on synchronous events. Initially the NFS server thread is blocked on the event. When the timeout expires, the thread unblocks and the file operation fails with an **ENOTREADY** error code. The event itself continues to exist and will eventually be handled. When a response for the event arrives at the source node it is saved. NFS is expected to periodically retry the operation. The retry will either find the response which has arrived between retries, or cause the operation to fail again with **ENOTREADY**. After repeated retries, the operation is eventually expected to succeed.

The interval is configurable using the **dmapiEventTimeout** configuration attribute on the **mmchconfig** command. See "GPFS configuration attributes for DMAPI" on page 579. The default is no timeout.

The timeout mechanism is activated only for data events (read, write, truncate), and only when the file operation comes from NFS.

Quota

GPFS supports user quota. When **dm_punch_hole** is invoked, the file owner's quota is adjusted by the disk space that is freed. The quota is also adjusted when **dm_write_invis** is invoked and additional disk space is consumed.

Since **dm_write_invis** runs with root credentials, it will never fail due to insufficient quota. However, it is possible that the quota of the file owner will be exceeded as a result of the invisible write. In that case the owner will not be able to perform further file operations that consume quota.

Memory mapped files

In GPFS, a read event or a write event will be generated (if enabled) at the time the memory mapping of a file is established.

No events will be generated during actual mapped access, regardless of the setting of the event list or the managed regions. Access to the file with regular file operations, while the file is memory mapped, will generate events, if such events are enabled.

To protect the integrity of memory mapped access, the DM application is not permitted to punch a hole in a file while the file is memory mapped. If the DM application calls **dm_punch_hole** while the file is memory mapped, the error code **EBUSY** will be returned.

Administration of IBM Spectrum Scale Data Management API for GPFS

To set up the DMAPI for GPFS, install the DMAPI files that are included in the GPFS installation package, and then choose the configuration attributes for DMAPI with the **mmchconfig** command. For each file system that you want DMAPI access, enable DMAPI with the **-z** flag of the **mmcrfs** or **mmchfs** command.

All DMAPI APIs must be called from nodes that are in the cluster where the file system is created. DMAPI APIs may **not** be invoked from a remote cluster. The GPFS daemon and each DMAPI application must be synchronized to prevent failures.

Administration of DMAPI for GPFS includes:

- "Required files for implementation of Data Management applications"
- "GPFS configuration attributes for DMAPI" on page 579
- "Enabling DMAPI for a file system" on page 580
- "Initializing the Data Management application" on page 581

Required files for implementation of Data Management applications

The installation image for GPFS contains the required files for implementation of Data Management applications.

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

The required files are:

dmapi.h
: The header file that contains the C declarations of the DMAPI functions.

  This header file must be included in the source files of the DM application.

  The file is installed in directory: **/usr/lpp/mmfs/include**.

dmapi_types.h
: The header file that contains the C declarations of the data types for the DMAPI functions and event messages.

  The header file **dmapi.h** includes this header file.

  The file is installed in directory: **/usr/lpp/mmfs/include**.

libdmapi.a
: The library that contains the DMAPI functions.

  The library **libdmapi.a** consists of a single shared object, which is built with auto-import of the system calls that are listed in the export file **dmapi.exp**.

  The file is installed in directory: **/usr/lpp/mmfs/lib**.

dmapi.exp
: The export file that contains the DMAPI system call names.

  The file **dmapi.exp** needs to be explicitly used only if the DM application is to be explicitly built with static binding, using the binder options **-bnso -bI:dmapi.exp**.

  The file is installed in directory: **/usr/lpp/mmfs/lib**.

dmapicalls, dmapicalls64
Module loaded during processing of the DMAPI functions.

The module is installed in directory: **/usr/lpp/mmfs/bin**.

Notes:

- On Linux nodes running DMAPI, the required files **libdmapi.a**, **dmapi.exp**, **dmapicalls**, and **dmapicalls64** are replaced by **libdmapi.so**.
- If you are compiling with a non-IBM compiler on AIX nodes, you must compile DMAPI applications with -D_AIX.

GPFS configuration attributes for DMAPI

GPFS uses several attributes for DMAPI that define various timeout intervals. These attributes can be changed with the **mmchconfig** command.

The DMAPI configuration attributes are:

dmapiDataEventRetry
Controls how GPFS handles the data event when it is enabled again right after this event is handled by the DMAPI application. Valid values are:

-1 Specifies that GPFS will always regenerate 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.

Positive Number
Specifies how many times the data event should be retried. The default is 2, which should be enough to cover most DMAPI applications. Unless a special situation occurs, you can increase this to a larger number or even set this to **-1** to always regenerate the events. Unless you perform careful testing, IBM recommends that you never change the default setting.

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 will block. 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 will find the response of the original event and continue. This mechanism applies only to read, write, and truncate events, 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. See also “Support for NFS” on page 577.

dmapiFileHandleSize
Controls the size of file handles generated by GPFS. The default DMAPI file handle size is 32 bytes. For clusters created prior to GPFS 3.2, the default DMAPI file handle size is 16 bytes.

Note: To change the DMAPI file handle size, GPFS must be stopped on all nodes in the cluster.

dmapiMountEvent
Controls the generation of the **mount**, **preunmount**, and **unmount** events. Valid values are:

all Specifies that **mount**, **preunmount**, and **unmount** events are generated on each node. This is the default behavior.

LocalNode
Specifies that **mount**, **preunmount**, and **unmount** events are generated only if the node is a session node.

SessionNode
Specifies that **mount**, **preunmount**, and **unmount** events are generated on each node and are delivered to the session node, but the session node will respond with **DM_RESP_CONTINUE** to the event node without delivering the event to the DMAPI application, unless the event is originated from the **SessionNode** itself.

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 mount of a file system which 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 will wait for the mount disposition. The parameter value is the maximum time, in seconds, that the mount operation will wait for a disposition. When this time expires and there still is 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. See also “Mount and unmount” on page 574 and “Initializing the Data Management application” on page 581.

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 suffered a failure. The parameter value is the maximum time, in seconds, the thread will wait for the recovery of the failed session. When this time expires and the session has not yet recovered, the event is aborted 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. See also “Failure and recovery of IBM Spectrum Scale Data Management API for GPFS” on page 601 for details on session failure and recovery.

Enabling DMAPI for a file system

DMAPI must be enabled individually for each file system.

DMAPI can be enabled for a file system when the file system is created, using the **-z yes** option on the **mmcrfs** command. The default is **-z no**. The setting can be changed when the file system is not mounted anywhere, using the **-z yes | no** option on the **mmchfs** command. The setting is persistent.

The current setting can be queried using the **-z** option on the **mmlsfs** command.

While DMAPI is disabled for a given file system, no events are generated by file operations of that file system. Any DMAPI function calls referencing that file system fail with an **EPERM** error.

When **mmchfs -z no** is used to disable DMAPI, existing event lists, extended attributes, and managed regions in the given file system remain defined, but will be ignored until DMAPI is re-enabled. The command **mmchfs -z no** should be used with caution, since punched holes, if any, are no longer protected by managed regions.

For more information about GPFS commands, see Chapter 1, “Command reference,” on page 1.

Initializing the Data Management application

All DMAPI APIs must be called from nodes that are in the cluster where the file system is created. DMAPI APIs may **not** be invoked from a remote cluster.

During initialization of GPFS, it is necessary to synchronize the GPFS daemon and the DM application to prevent mount operations from failing. There are two mechanisms to accomplish this:

1. The shell script **gpfsready** invoked by the GPFS daemon during initialization.
2. A timeout interval, allowing mount operations to wait for a disposition to be set for the mount event.

During GPFS initialization, the daemon invokes the shell script **gpfsready**, located in directory **/var/mmfs/etc**. This occurs as the file systems are starting to be mounted. The shell script can be modified to start or restart the DM application. Upon return from this script, a session should have been created and a disposition set for the mount event. Otherwise, mount operations may fail due to a lack of disposition.

In a multiple-node environment such as GPFS, usually only a small subset of the nodes are session nodes, having DM applications running locally. On a node that is not a session node, the **gpfsready** script can be modified to synchronize between the local GPFS daemon and a remote DM application. This will prevent mount from failing on any node.

A sample shell script **gpfsready.sample** is installed in directory **/usr/lpp/mmfs/samples**.

If no mount disposition has ever been set in the cluster, the first external mount of a DMAPI-enabled file system on each node will activate a timeout interval on that node. Any mount operation on that node that starts during the timeout interval will wait for the mount disposition until the timeout expires. The timeout interval is configurable using the **dmapiMountTimeout** configuration attribute on the **mmchconfig** command (the interval can even be made infinite). A message is displayed at the beginning of the wait. If there is still no disposition for the mount event when the timeout expires, the mount operation will fail with an **EIO** error code. See "GPFS configuration attributes for DMAPI" on page 579 for more information on **dmapiMountTimeout**.

Specifications of enhancements for IBM Spectrum Scale Data Management API for GPFS

DMAPI for GPFS provides numerous enhancements in data structures and functions.

These enhancements are provided mainly by the multiple-node environment. Some data structures have additional fields. Many functions have usage restrictions, changes in semantics, and additional error codes. The enhancements are in these areas:

- "Enhancements to data structures"
- "Usage restrictions on DMAPI functions" on page 583
- "Definitions for GPFS-specific DMAPI functions" on page 584
- "Semantic changes to DMAPI functions" on page 598
- "GPFS-specific DMAPI events" on page 599
- "Additional error codes returned by DMAPI functions" on page 600

Enhancements to data structures

This is a description of GPFS enhancements to data structures defined in the XDSM standard.

For complete C declarations of all the data structures that are used in DMAPI for GPFS, refer to the **dmapi_types.h** file located in the **/usr/lpp/mmfs/include** directory.

- All file offsets and sizes in DMAPI data structures are 64 bits long.

- Names or path names that are passed in event messages are character strings, terminated by a null character. The length of the name buffer, as specified in the **dm_vardata_t** structure, includes the null character.
- The **dm_region_t** structure has a new 4-byte field, **rg_opaque**. The DMAPI implementation does not interpret **rg_opaque**. The DM application can use this field to store additional information within the managed region.
- The **dt_change** field in the **dm_stat** structure is not implemented in the inode. The value will change each time it is returned by the **dm_get_fileattr** function.
- The **dt_dtime** field in the **dm_stat** structure is overloaded on the **dt_ctime** field.
- The **dm_eventmsg** structure has a 4 byte field, **ev_nodeid** that uniquely identifies the node that generated the event. The id is the GPFS cluster data node number, which is attribute **node_number** in the **mmsdrfs2** file for a PSSP node or **mmsdrfs** file for any other type of node.
- The **ne_mode** field in the **dm_namesp_event** structure has an additional flag, **DM_LOCAL_MOUNT**. For the events preunmount and unmount when this flag is set, the unmount operation is local to the session node. See "Mount and unmount" on page 574. The **me_mode** field in the **dm_mount_event** structure has two additional flags; **DM_LOCAL_MOUNT**, and **DM_REMOTE_MOUNT**. See "Mount and unmount" on page 574.
- There are two 'quick access' single-bit opaque DM attributes for each file, stored directly in the inode. See "Data Management attributes" on page 577.
- The data type **dm_eventset_t** is implemented as a bit map, containing one bit for each event that is defined in DMAPI. The bit is set if, and only if, the event is present.

  Variables of type **dm_eventset_t** should be manipulated only using special macros. The XDSM standard provides a basic set of such macros. GPFS provides a number of additional macros. The names of all such macros begin with the prefix **DMEV_**.

  This is the list of additional macros that are provided in DMAPI for GPFS:

  DMEV_ALL(eset)
  Add all events to **eset**

  DMEV_ISZERO(eset)
  Check if **eset** is empty

  DMEV_ISALL(eset)
  Check if **eset** contains all events

  DMEV_ADD(eset1, eset2)
  Add to **eset2** all events in **eset1**

  DMEV_REM(eset1, eset2)
  Remove from **eset2** all events in **eset1**

  DMEV_RES(eset1, eset2)
  Restrict **eset2** by **eset1**

  DMEV_ISEQ(eset1, eset2)
  Check if **eset1** and **eset2** are equal

  DMEV_ISDISJ(eset1, eset2)
  Check if **eset1** and **eset2** are disjoint

  DMEV_ISSUB(eset2)
  Check if **eset1** is a subset of **eset2**

  DMEV_NORM(eset)
  Normalize the internal format of **eset**, clearing all unused bits
- DMAPI for GPFS provides a set of macros for comparison of token ids (value of type **dm_token_t**).

  DM_TOKEN_EQ (x,y)
  Check if **x** and **y** are the same

DM_TOKEN_NE (x,y)
Check if **x** and **y** are different

DM_TOKEN_LT (x,y)
Check if **x** is less than **y**

DM_TOKEN_GT (x,y)
Check if **x** is greater than **y**

DM_TOKEN_LE (x,y)
Check if **x** is less than or equal to **y**

DM_TOKEN_GE (x,y)
Check if **x** is greater than or equal to **y**

Usage restrictions on DMAPI functions

There are usage restrictions on the DMAPI for GPFS functions.

- The maximum number of DMAPI sessions that can be created on a node is 4000.
- Root credentials are a prerequisite for invoking any DMAPI function, otherwise the function fails with an **EPERM** error code.
- DMAPI functions are unable to run if the GPFS kernel extension is not loaded, or if the runtime module **dmapicalls** is not installed. An **ENOSYS** error code is returned in this case.
- Invoking a DMAPI function that is not implemented in GPFS results in returning the **ENOSYS** error code.
- DMAPI functions will fail, with the **ENOTREADY** error code, if the local GPFS daemon is not running.
- DMAPI functions will fail, with the **EPERM** error code, if DMAPI is disabled for the file system that is referenced by the file handle argument.
- DMAPI functions cannot access GPFS reserved files, such as quota files, inode allocation maps, and so forth. The **EBADF** error code is returned in this case.
- GPFS does not support access rights on entire file systems (as opposed to individual files). Hence, DMAPI function calls that reference a file system (with a file system handle) cannot present a token, and must use **DM_NO_TOKEN**. Functions affected by this restriction are:
  - **dm_set_eventlist**
  - **dm_get_eventlist**
  - **dm_set_disp**
  - **dm_get_mountinfo**
  - **dm_set_return_on_destroy**
  - **dm_get_bulkattr**
  - **dm_get_bulkall**

  If a token is presented, these functions fail with the **EINVAL** error code.
- DMAPI functions that acquire, change, query, or release access rights, must not present a file system handle. These functions are:
  - **dm_request_right**
  - **dm_upgrade_right**
  - **dm_downgrade_right**
  - **dm_release_right**
  - **dm_query_right**

  If a file system handle is presented, these functions fail with the **EINVAL** error code.
- The function **dm_request_right**, when invoked without wait (the *flags* argument has a value of 0), will almost always fail with the **EAGAIN** error. A GPFS implementation constraint prevents this function

from completing successfully without wait, even if it is known that the requested access right is available. The **DM_RR_WAIT** flag must always be used. If the access right is available, there will be no noticeable delay.

- DMAPI function calls that reference a specific token, either as input or as output, can be made only on the session node. Otherwise, the call fails with the **EINVAL** error code.
- DMAPI function calls that reference an individual file by handle must be made on the session node. The corresponding file system must be mounted on the session node. The call fails with **EINVAL** if it is not on the session node, and with **EBADF** if the file system is not mounted.
- DMAPI function calls that reference a file system by handle (as opposed to an individual file) can be made on any node, not just the session node. The relevant functions are:
  - **dm_set_eventlist**
  - **dm_get_eventlist**
  - **dm_set_disp**
  - **dm_get_mountinfo**
  - **dm_set_return_on_destroy**
  - **dm_get_bulkattr**
  - **dm_get_bulkall**

  For **dm_get_bulkattr** and **dm_get_bulkall**, the system file must be mounted on the node that is making the call. For the other functions, the file system must be mounted on some node, but not necessarily on the node that is making the call. As specified previously, all such function calls must use **DM_NO_TOKEN**. The function fails with the **EBADF** error code if the file system is not mounted as required.
- The function **dm_punch_hole** will fail with the **EBUSY** error code if the file to be punched is currently memory-mapped.
- The function **dm_move_event** can only be used when the source session and the target session are on the same node. The function must be called on the session node. Otherwise, the function fails with the **EINVAL** error code.
- The function **dm_create_session**, when providing an existing session id in the argument *oldsid*, can only be called on the session node, except after session node failure. Otherwise, the call will return the **EINVAL** error code.
- The function **dm_destroy_session** can only be called on the session node, otherwise the call will fail with the **EINVAL** error code.
- The function **dm_set_fileattr** cannot change the file size. If the **dm_at_size** bit in the attribute mask is set, the call fails with the **EINVAL** error code.
- DMAPI functions that reference an event with a token fail with the **ESRCH** error code, if the event is not in an outstanding state. This is related to session recovery. See "Failure and recovery of IBM Spectrum Scale Data Management API for GPFS" on page 601 for details on session failure and recovery.

For additional information about:

- Semantic changes to the DMAPI for GPFS functions, see "Semantic changes to DMAPI functions" on page 598.
- C declarations of all functions in DMAPI for GPFS, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory.

Definitions for GPFS-specific DMAPI functions

The GPFS-specific DMAPI functions are not part of the DMAPI open standard.

You can use the following GPFS-specific DMAPI functions to work with file system snapshots:

- "dm_handle_to_snap" on page 586

- "dm_make_xhandle" on page 587

You can use the following GPFS-specific DMAPI functions to make asynchronous updates to attributes, managed regions, and event lists on files:

- "dm_remove_dmattr_nosync" on page 589
- "dm_set_dmattr_nosync" on page 591
- "dm_set_eventlist_nosync" on page 593
- "dm_set_region_nosync" on page 595

You can use the following GPFS-specific DMAPI function to make the previously listed asynchronous updates persistent by flushing them to disk:

- "dm_sync_dmattr_by_handle" on page 597

dm_handle_to_snap

Extracts a snapshot ID from a handle.

Synopsis

```
int dm_handle_to_snap(
    void          *hanp,          /* IN */
    size_t        hlen,           /* IN */
    dm_snap_t     *isnapp         /* OUT */
);
```

Description

Use the **dm_handle_to_snap** function to extract a snapshot ID from a handle. **dm_handle_to_snap()** is a GPFS-specific DMAPI function. It is not part of the open standard.

Parameters

void *_hanp_ (IN)
A pointer to an opaque DM handle previously returned by DMAPI.

size_t _hlen_ (IN)
The length of the handle in bytes.

dm_snap_t *_isnapp_ (OUT)
A pointer to the snapshot ID.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[ENOMEM]
DMAPI could not obtain the required resources to complete the call.

[ENOSYS]
Function is not supported by the DM implementation.

[EPERM]
The caller does not hold the appropriate privilege.

See also

"dm_make_xhandle" on page 587

dm_make_xhandle

Converts a file system ID, inode number, inode generation count, and snapshot ID into a handle.

Synopsis

```
int dm_make_xhandle(
    dm_fsid_t               *fsidp,         /* IN */
    dm_ino_t                *inop,          /* IN */
    dm_igen_t               *igenp,         /* IN */
    dm_snap_t               *isnapp,        /* IN */
    void                    **hanpp,        /* OUT */
    size_t                  *hlenp          /* OUT */
);
```

Description

Use the **dm_make_xhandle()** function to convert a file system ID, inode number, inode generation count, and snapshot ID into a handle. **dm_make_xhandle()** is a GPFS-specific DMAPI function. It is not part of the open standard.

Parameters

dm_fsid_t *_fsidp_ (IN)
The file system ID.

dm_ino_t *_inop_ (IN)
The inode number.

dm_igen_t *_igenp_ (IN)
The inode generation count.

dm_snap_t *_isnapp_ (IN)
The snapshot ID.

void **_hanpp_ (OUT)
A DMAPI initialized pointer that identifies a region of memory containing an opaque DM handle. The caller is responsible for freeing the allocated memory.

size_t *_hlenp_ (OUT)
The length of the handle in bytes.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[ENOMEM]
DMAPI could not obtain the required resources to complete the call.

[ENOSYS]
Function is not supported by the DM implementation.

[EPERM]
The caller does not hold the appropriate privilege.

See also

"dm_handle_to_snap" on page 586

dm_remove_dmattr_nosync

Asynchronously removes the specified attribute.

Synopsis

```
int dm_remove_dmattr_nosync(
    dm_sessid_t      sid,
    void            *hanp,
    size_t           hlen,
    dm_token_t       token,
    int              setdtime,
    dm_attrname_t   *attrnamep
);
```

Description

Use the **dm_remove_dmattr_nosync** function to asynchronously remove the attribute specified by *attrname*.

dm_remove_dmattr_nosync is a GPFS-specific DMAPI function; it is not part of the open standard. It has the same purpose, parameters, and return values as the standard DMAPI **dm_remove_dmattr** function, except that the update that it performs is not persistent until some other activity on that file (or on other files in the file system) happens to flush it to disk. To be certain that your update is made persistent, use one of the following functions:

- Standard DMAPI **dm_sync_by_handle** function, which flushes the file data and attributes
- GPFS-specific **dm_sync_dmattr_by_handle** function, which flushes only the attributes.

Parameters

dm_sessid_t *sid* (IN)
The identifier for the session of interest.

void **hanp* (IN)
The handle for the file for which the attributes should be removed.

size_t *hlen* (IN)
The length of the handle in bytes.

dm_token_t **token* (IN)
The token referencing the access right for the handle. The access right must be **DM_RIGHT_EXCL**, or the token **DM_NO_TOKEN** may be used and the interface acquires the appropriate rights.

int *setdtime* (IN)
If *setdtime* is non-zero, updates the file's attribute time stamp.

dm_attrname_t **attrnamep* (IN)
The attribute to be removed.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[EACCES]
The access right referenced by the token for the handle is not **DM_RIGHT_EXCL**.

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[EINVAL]
The session is not valid.

[EIO] I/O error resulted in failure of operation.

[ENOSYS]
The DMAPI implementation does not support this optional function.

[EPERM]
The caller does not hold the appropriate privilege.

[EROFS]
The operation is not allowed on a read-only file system.

See also

"dm_set_dmattr_nosync" on page 591, "dm_sync_dmattr_by_handle" on page 597

dm_set_dmattr_nosync

Asynchronously creates or replaces the value of the named attribute with the specified data.

Synopsis

```
int dm_set_dmattr_nosync(
    dm_sessid_t     sid,
    void           *hanp,
    size_t          hlen,
    dm_token_t      token,
    dm_attrname_t  *attrnamep,
    int             setdtime,
    size_t          buflen,
    void           *bufp
);
```

Description

Use the **dm_set_dmattr_nosync** function to asynchronously create or replace the value of the named attribute with the specified data.

dm_set_dmattr_nosync is a GPFS-specific DMAPI function; it is not part of the open standard. It has the same purpose, parameters, and return values as the standard DMAPI **dm_set_dmattr** function, except that the update that it performs is not persistent until some other activity on that file (or on other files in the file system) happens to flush it to disk. To be certain that your update is made persistent, use one of the following functions:

- Standard DMAPI **dm_sync_by_handle** function, which flushes the file data and attributes
- GPFS-specific **dm_sync_dmattr_by_handle** function, which flushes only the attributes.

Parameters

dm_sessid_t *sid* (IN)
The identifier for the session of interest.

void *hanp* (IN)
The handle for the file for which the attributes should be created or replaced.

size_t *hlen* (IN)
The length of the handle in bytes.

dm_token_t *token* (IN)
The token referencing the access right for the handle. The access right must be **DM_RIGHT_EXCL**, or the token **DM_NO_TOKEN** may be used and the interface acquires the appropriate rights.

dm_attrname_t *attrnamep* (IN)
The attribute to be created or replaced.

int *setdtime* (IN)
If *setdtime* is non-zero, updates the file's attribute time stamp.

size_t *buflen* (IN)
The size of the buffer in bytes.

void *bufp* (IN)
The buffer containing the attribute data.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[E2BIG]
The attribute value exceeds one of the implementation defined storage limits.

[E2BIG]
buflen is larger than the implementation defined limit. The limit can be determined by calling the **dm_get_config()** function.

[EACCES]
The access right referenced by the token for the handle is not **DM_RIGHT_EXCL**.

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EIO] An attempt to write the new or updated attribute resulted in an I/O error.

[EINVAL]
The argument *token* is not a valid token.

[EINVAL]
The session is not valid.

[ENOMEM]
The DMAPI could not acquire the required resources to complete the call.

[ENOSPC]
An attempt to write the new or updated attribute resulted in an error due to no free space being available on the device.

[ENOSYS]
The DMAPI implementation does not support this optional function.

[EPERM]
The caller does not hold the appropriate privilege.

[EROFS]
The operation is not allowed on a read-only file system.

See also

"dm_remove_dmattr_nosync" on page 589, "dm_sync_dmattr_by_handle" on page 597

dm_set_eventlist_nosync

Asynchronously sets the list of events to be enabled for an object.

Synopsis

```
int dm_set_eventlist_nosync(
    dm_sessid_t      sid,
    void            *hanp,
    size_t           hlen,
    dm_token_t       token,
    dm_eventset_t   *eventsetp,
    u_int            maxevent
);
```

Description

Use the **dm_set_eventlist_nosync** function to asynchronously set the list of events to be enabled for an object.

dm_set_eventlist_nosync is a GPFS-specific DMAPI function; it is not part of the open standard. It has the same purpose, parameters, and return values as the standard DMAPI **dm_set_eventlist** function, except that the update that it performs is not persistent until some other activity on that file (or on other files in the file system) happens to flush it to disk. To be certain that your update is made persistent, use one of the following functions:

- Standard DMAPI **dm_sync_by_handle** function, which flushes the file data and attributes
- GPFS-specific **dm_sync_dmattr_by_handle** function, which flushes only the attributes.

Parameters

dm_sessid_t *sid* (IN)
The identifier for the session of interest.

void **hanp* (IN)
The handle for the object. The handle can be either the system handle or a file handle.

size_t *hlen* (IN)
The length of the handle in bytes.

dm_token_t **token* (IN)
The token referencing the access right for the handle. The access right must be **DM_RIGHT_EXCL**, or the token **DM_NO_TOKEN** may be used and the interface acquires the appropriate rights.

dm_eventset_t **eventsetp* (IN)
The list of events to be enabled for the object.

u_int *maxevent* (IN)
The number of events to be checked for dispositions in the event set. The events from 0 to *maxevent***-1** are examined.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[EACCES]
The access right referenced by the token for the handle is not **DM_RIGHT_EXCL**.

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[EINVAL]
The session is not valid.

[EINVAL]
Tried to set event on a global handle.

[ENOMEM]
The DMAPI could not acquire the required resources to complete the call.

[ENXIO]
The implementation of the DMAPI does not support enabling event delivery on the specified handle.

[EPERM]
The caller does not hold the appropriate privilege.

[EROFS]
The operation is not allowed on a read-only file system.

See also

"dm_sync_dmattr_by_handle" on page 597

dm_set_region_nosync

Asynchronously replaces the set of managed regions for a file.

Synopsis

```
int dm_set_region_nosync(
    dm_sessid_t     sid,
    void           *hanp,
    size_t          hlen,
    dm_token_t      token,
    u_int           nelem,
    dm_region_t    *regbufp,
    dm_boolean_t   *exactflagp
);
```

Description

Use the **dm_set_region_nosync** function to asynchronously replace the set of managed regions for a file.

dm_set_region_nosync is a GPFS-specific DMAPI function; it is not part of the open standard. It has the same purpose, parameters, and return values as the standard DMAPI **dm_set_region** function, except that the update that it performs is not persistent until some other activity on that file (or on other files in the file system) happens to flush it to disk. To be certain that your update is made persistent, use one of the following functions:

- Standard DMAPI **dm_sync_by_handle** function, which flushes the file data and attributes
- GPFS-specific **dm_sync_dmattr_by_handle** function, which flushes only the attributes.

Parameters

dm_sessid_t *sid* (IN)
The identifier for the session of interest.

void **hanp* (IN)
The handle for the regular file to be affected.

size_t *hlen* (IN)
The length of the handle in bytes.

dm_token_t **token* (IN)
The token referencing the access right for the handle. The access right must be **DM_RIGHT_EXCL**, or the token **DM_NO_TOKEN** may be used and the interface acquires the appropriate rights.

u_int *nelem* (IN)
The number of input regions in *regbufp*. If *nelem* is 0, then all existing managed regions are cleared.

dm_region_t **regbufp* (IN)
A pointer to the structure defining the regions to be set. May be NULL if *nelem* is zero.

dm_boolean_t **exactflagp* (OUT)
If **DM_TRUE**, the file system did not alter the requested managed region set.

Valid values for the *rg_flags* field of the region structure are created by OR'ing together one or more of the following values:

DM_REGION_READ
Enable synchronous event for read operations that overlap this managed region.

DM_REGION_WRITE
Enable synchronous event for write operations that overlap this managed region.

DM_REGION_TRUNCATE
Enable synchronous event for truncate operations that overlap this managed region.

DM_REGION_NOEVENT
Do not generate any events for this managed region.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[E2BIG]
The number of regions specified by nelem exceeded the implementation capacity.

[EACCES]
The access right referenced by the token for the handle is not **DM_RIGHT_EXCL**.

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[EINVAL]
The file handle does not refer to a regular file.

[EINVAL]
The regions passed in are not valid because they overlap or some other problem.

[EINVAL]
The session is not valid.

[EIO] An I/O error resulted in failure of operation.

[ENOMEM]
The DMAPI could not acquire the required resources to complete the call.

[EPERM]
The caller does not hold the appropriate privilege.

[EROFS]
The operation is not allowed on a read-only file system.

See also

"dm_sync_dmattr_by_handle" on page 597

dm_sync_dmattr_by_handle

Synchronizes one or more files' in-memory attributes with those on the physical medium.

Synopsis

```
int m_sync_dmattr_by_handle(
    dm_sessid_t    sid,
    void          *hanp,
    size_t         hlen,
    dm_token_t     token
);
```

Description

Use the **dm_sync_dmattr_by_handle** function to synchronize one or more files' in-memory attributes with those on the physical medium.

dm_sync_dmattr_by_handle is a GPFS-specific DMAPI function; it is not part of the open standard. It has the same purpose, parameters, and return values as the standard DMAPI **dm_sync_by_handle** function, except that it flushes only the attributes, not the file data.

Like **dm_sync_by_handle**, **dm_sync_dmattr_by_handle** commits all previously unsynchronized updates for that node, not just the updates for one file. Therefore, if you update a list of files and call **dm_sync_dmattr_by_handle** on the last file, the attribute updates to all of the files in the list are made persistent.

Parameters

dm_sessid_t *sid* (IN)
The identifier for the session of interest.

void **hanp* (IN)
The handle for the file whose attributes are to be synchronized.

size_t *hlen* (IN)
The length of the handle in bytes.

dm_token_t **token* (IN)
The token referencing the access right for the handle. The access right must be **DM_RIGHT_EXCL**, or the token **DM_NO_TOKEN** may be used and the interface acquires the appropriate rights.

Return values

Zero is returned on success. On error, -1 is returned, and the global *errno* is set to one of the following values:

[EACCES]
The access right referenced by the token for the handle is not **DM_RIGHT_EXCL**.

[EBADF]
The file handle does not refer to an existing or accessible object.

[EFAULT]
The system detected an invalid address in attempting to use an argument.

[EINVAL]
The argument *token* is not a valid token.

[ENOMEM]
The DMAPI could not acquire the required resources to complete the call.

[ENOSYS]
The DMAPI implementation does not support this optional function.

[EPERM]
The caller does not hold the appropriate privilege.

See also

"dm_remove_dmattr_nosync" on page 589, "dm_set_dmattr_nosync" on page 591, "dm_set_eventlist_nosync" on page 593, and "dm_set_region_nosync" on page 595

Semantic changes to DMAPI functions

There are semantic changes to functions in DMAPI for GPFS. These changes are entailed mostly by the multiple-node environment.

For a list of additional error codes that are used in DMAPI for GPFS, see "Additional error codes returned by DMAPI functions" on page 600. For C declarations of all the DMAPI for GPFS functions, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory.

- The following DMAPI functions can be invoked on any node, not just the session node, as long as the session exists on some node in the GPFS cluster.
  - **dm_getall_disp**
  - **dm_query_session**
  - **dm_send_msg**
- DMAPI functions that reference a file system, as opposed to an individual file, can be made on any node, not just the session node. Being able to call certain functions on any node has advantages. The DM application can establish event monitoring when receiving a mount event from any node. Also, a distributed DM application can change event lists and dispositions of any file system from any node.
  - **dm_set_eventlist**
  - **dm_get_eventlist**
  - **dm_set_disp**
  - **dm_get_mount_info**
  - **dm_set_return_on_destroy**
  - **dm_get_bulkattr**
  - **dm_get_bulkall**
- The following functions, that construct a handle from its components, do not check if the resulting handle references a valid file. Validity is checked when the handle is presented in function calls that actually reference the file.
  - **dm_make_handle**
  - **dm_make_fshandle**
  - **dm_make_xhandle**
- The following data movement functions may be invoked on any node within the GPFS cluster, provided they are run as root and present a session ID for an established session on the session node. For guidelines on how to perform data movement from multiple nodes, see "Parallelism in Data Management applications" on page 576.
  - **dm_read_invis**
  - **dm_write_invis**
  - **dm_probe_hole**
  - **dm_punch_hole**

- The following functions that extract components of the handle, do not check whether the specified handle references a valid file. Validity is checked when the handle is presented in function calls that actually reference the file.
  - **dm_handle_to_fsid**
  - **dm_handle_to_igen**
  - **dm_handle_to_ino**
  - **dm_handle_to_snap**
- **dm_handle_to_fshandle** converts a file handle to a file system handle without checking the validity of either handle.
- **dm_handle_is_valid** does not check if the handle references a valid file. It verifies only that the internal format of the handle is correct.
- **dm_init_attrloc** ignores all of its arguments, except the output argument *locp*. In DMAPI for GPFS, the location pointer is initialized to a constant. Validation of the session, token, and handle arguments is done by the bulk access functions.
- When **dm_query_session** is called on a node other than the session node, it returns only the first eight bytes of the session information string.
- **dm_create_session** can be used to move an existing session to another node, if the current session node has failed. The call must be made on the new session node. See "Failure and recovery of IBM Spectrum Scale Data Management API for GPFS" on page 601 for details on session node failure and recovery.
- Assuming an existing session, using **dm_create_session** does not change the session id. If the argument *sessinfop* is **NULL**, the session information string is not changed.
- The argument *maxevent* in the functions **dm_set_disp** and **dm_set_eventlist** is ignored. In GPFS the set of events is implemented as a bitmap, containing a bit for each possible event.
- The value pointed to by the argument *nelemp*, on return from the functions **dm_get_eventlist** and **dm_get_config_events**, is always **DM_EVENT_MAX-1**. The argument *nelem* in these functions is ignored.
- The *dt_nevents* field in the **dm_stat_t** structure, which is returned by the **dm_get_fileattr** and **dm_get_bulkall** functions, has a value of **DM_EVENT_MAX-1** when the file has a file-system–wide event enabled by calling the **dm_set_eventlist** function. The value will always be **3** when there is no file-system–wide event enabled. A value of **3** indicates that there could be a managed region enabled for the specific file, which might have enabled a maximum of three events: READ, WRITE, and TRUNCATE.
- The functions **dm_get_config** and **dm_get_config_events** ignore the arguments *hanp* and *hlen*. This is because the configuration is not dependent on the specific file or file system.
- The function **dm_set_disp**, when called with the global handle, ignores any events in the event set being presented, except the mount event. When **dm_set_disp** is called with a file system handle, it ignores the mount event.
- The function **dm_handle_hash**, when called with an individual file handle, returns the inode number of the file. When **dm_handle_hash** is called with a file system handle, it returns the value 0.
- The function **dm_get_mountinfo** returns two additional flags in the **me_mode** field in the **dm_mount_event** structure. The flags are **DM_MOUNT_LOCAL** and **DM_MOUNT_REMOTE**. See "Mount and unmount" on page 574 for details.

GPFS-specific DMAPI events

The GPFS-specific events are not part of the DMAPI open standard. You can use these GPFS events to filter out events that are not critical to file management and to prevent system overloads from trivial information.

The DMAPI standard specifies that the system must generate ATTRIBUTE events each time the "changed time" (**ctime**) attribute for a file changes. For systems that write files in parallel, like GPFS, this generates ATTRIBUTE events from every node writing to the file. Consequently, it is easy for ATTRIBUTE events to

overwhelm a data management server. However, the only **ctime** changes that are critical to GPFS are changes to either the permissions or ACLs of a file. In most cases, GPFS can ignore other **ctime** changes.

To distinguish file permission and ACL changes from other **ctime** updates, the following DMAPI metadata attribute events allow GPFS to filter **ctime** updates. Using these events, DM servers are able to track file permission changes without overwhelming the system with irrelevant ATTRIBUTE events. However, these events are not part of the CAE Specification C429 open standard and they were implemented specifically for GPFS systems.

Metadata Events

DM_EVENT_PREPERMCHANGE
Pre-permission change event. Event is triggered before file permission change.

DM_EVENT_POSTPERMCHANGE
Post-permission change event. Event is triggered after file permission change.

Note: If you only want to track permission and ACL changes, turn off the **DM_EVENT_ATTRIBUTE** and turn on both the **DM_EVENT_PREPERMCHANGE** and **DM_EVENT_POSTPERMCHANGE** events.

Additional error codes returned by DMAPI functions

DMAPI for GPFS uses additional error codes, not specified in the XDSM standard, for most DMAPI functions.

For C declarations of all the DMAPI for GPFS functions, refer to the **dmapi.h** file located in the **/usr/lpp/mmfs/include** directory.

For **all DMAPI functions**, these error codes are used:

ENOSYS
The GPFS kernel extension is not loaded, or the runtime module **dmapicalls** is not installed.

ENOSYS
An attempt has been made to invoke a DMAPI function that is not implemented in GPFS.

ENOTREADY
The local GPFS daemon is not running or is initializing.

ENOMEM
DMAPI could not acquire the required resources to complete the call. **ENOMEM** is defined in the XDSM standard for some DMAPI functions, but not for all.

ESTALE
An error has occurred which does not fit any other error code specified for this function.

For **DMAPI functions that provide a file handle as an input argument**, these error codes are used:

EINVAL
The format of the file handle is not valid.

This error is returned without attempting to locate any object that is referenced by the handle. The **EINVAL** error code is to be distinguished from the **EBADF** error code, which, as specified in the XDSM standard, indicates that the object does not exist or is inaccessible. Thus, GPFS provides a refinement, distinguishing between format and access errors related to handles.

EPERM
DMAPI is disabled for the file system that is referenced by the file handle.

For **DMAPI functions that provide a token as an input argument**, these error codes are used:

ESRCH
The event referenced by the token is not in outstanding state.

This is to be distinguished from the **EINVAL** error code, which is returned when the token itself is not valid. **ESRCH** is defined in the XDSM standard for some DMAPI functions, but not for all relevant functions. In GPFS, the **ESRCH** error code occurs mostly after recovery from session failure. See "Event recovery" on page 603 for details.

For these **specific DMAPI functions**, the error code listed is used:

Table 17. Specific DMAPI functions and associated error codes.

Name of function	Error codes and descriptions
dm_downgrade_right() **dm_upgrade_right()**	**EINVAL** - The session or token is not valid.
dm_get_region()	**EPERM** - The caller does not hold the appropriate privilege.
dm_init_service()	**EFAULT** - The system detected an invalid address in attempting to use an argument.
dm_move_event() **dm_respond_event()**	**EINVAL** - The token is not valid.
dm_punch_hole()	**EBUSY** - The file is currently memory mapped.
dm_probe_hole() **dm_punch_hole()**	**EINVAL** - The argument *len* is too large, and will overflow if cast into **offset_t**. **EINVAL** - The argument *off* is negative.
dm_write_invis()	**EINVAL** - The argument *flags* is not valid.
dm_read_invis() **dm_write_invis()**	**EINVAL** - The argument *len* is too large, and will overflow if placed into the **uio_resid** field in the structure **uio**. **EINVAL** - The argument *off* is negative.
dm_sync_by_handle()	**EROFS** - The operation is not allowed on a read-only file system.
dm_find_eventmsg() **dm_get_bulkall()** **dm_get_bulkattr()** **dm_get_dirattrs()** **dm_get_events()** **dm_get_mountinfo()** **dm_getall_disp()** **dm_getall_dmattr()** **dm_handle_to_path()**	**EINVAL** - The argument *buflen* is too large; it must be smaller than **INT_MAX**.
dm_get_alloc_info() **dm_getall_sessions()** **dm_getall_tokens()**	**EINVAL** - The argument *nelem* is too large; DMAPI cannot acquire sufficient resources.

Failure and recovery of IBM Spectrum Scale Data Management API for GPFS

Failure and recovery of DMAPI applications in the multiple-node GPFS environment is different than in a single-node environment.

The failure model in XDSM is intended for a single-node environment. In this model, there are two types of failures:

DM application failure
The DM application has failed, but the file system works normally. Recovery entails restarting the DM application, which then continues handling events. Unless the DM application recovers, events may remain pending indefinitely.

Total system failure
The file system has failed. All non-persistent DMAPI resources are lost. The DM application itself may or may not have failed. Sessions are not persistent, so recovery of events is not necessary. The file system cleans its state when it is restarted. There is no involvement of the DM application in such cleanup.

The simplistic XDSM failure model is inadequate for GPFS. In a multiple-node environment, GPFS can fail on one node, but survive on other nodes. This type of failure is called *single-node failure* (*or partial system failure*). GPFS is built to survive and recover from single-node failures, without meaningfully affecting file access on surviving nodes.

Designers of Data Management applications for GPFS must comply with the enhanced DMAPI failure model, in order to support recoverability of GPFS. These areas are addressed:

- "Single-node failure"
- "Session failure and recovery" on page 603
- "Event recovery" on page 603
- "Loss of access rights" on page 604
- "DODeferred deletions" on page 604
- "DM application failure" on page 604

Single-node failure

In DMAPI for GPFS, single-node failure means that DMAPI resources are lost on the failing node, but not on any other node.

The most common single-node failure is when the local GPFS daemon fails. This renders any GPFS file system at that node inaccessible. Another possible single-node failure is file system forced unmount. When just an individual file system is forced unmounted on some node, its resources are lost, but the sessions on that node, if any, survive.

Single-node failure has a different effect when it occurs on a session node or on a source node:

session node failure
When the GPFS daemon fails, all session queues are lost, as well as all nonpersistent local file system resources, particularly DM access rights. The DM application may or may not have failed. The missing resources may in turn cause DMAPI function calls to fail with errors such as **ENOTREADY** or **ESRCH**.

Events generated at other source nodes remain pending despite any failure at the session node. Moreover, client threads remain blocked on such events.

source node failure
Events generated by that node are obsolete. If such events have already been enqueued at the session node, the DM application will process them, even though this may be redundant since no client is waiting for the response.

According to the XDSM standard, sessions are not persistent. This is inadequate for GPFS. Sessions must be persistent to the extent of enabling recovery from single-node failures. This is in compliance with a basic GPFS premise that single-node failures do not affect file access on surviving nodes. Consequently, after session node failure, the session queue and the events on it must be reconstructed, possibly on another node.

Session recovery is triggered by the actions of the DM application. The scenario depends on whether or not the DM application itself has failed.

If the DM application has failed, it must be restarted, possibly on another node, and assume the old session by id. This will trigger reconstruction of the session queue and the events on it, using backup information replicated on surviving nodes. The DM application may then continue handling events. The session id is never changed when a session is assumed.

If the DM application itself survives, it will notice that the session has failed by getting certain error codes from DMAPI function calls (**ENOTREADY**, **ESRCH**). The application could then be moved to another node and recover the session queue and events on it. Alternatively, the application could wait for the GPFS daemon to recover. There is also a possibility that the daemon will recover before the DM application even notices the failure. In these cases, session reconstruction is triggered when the DM application invokes the first DMAPI function after daemon recovery.

Session failure and recovery

A session fails when the GPFS daemon of the session node fails.

Session failure results in the loss of all DM access rights associated with events on the queue, and all the tokens become invalid. After the session has recovered, any previously outstanding synchronous events return to the initial (non-outstanding) state, and must be received again.

Session failure may also result in partial loss of the session information string. In such case, GPFS will be able to restore only the first eight characters of the session string. It is suggested to not have the DM application be dependent on more than eight characters of the session string.

In extreme situations, failure may also result in the loss of event dispositions for some file system. This happens only if the GPFS daemon fails simultaneously on all nodes where the file system was mounted. When the file system is remounted, a mount event will be generated, at which point the dispositions could be reestablished by the DM application.

During session failure, events originating from surviving nodes remain pending, and client threads remain blocked on such events. It is therefore essential that the DM application assume the old session and continue processing the pending events. To prevent indefinite blocking of clients, a mechanism has been implemented whereby pending events will be aborted and corresponding file operations failed with the **EIO** error if the failed session is not recovered within a specified time-out interval. The interval is configurable using the **dmapiSessionFailureTimeout** attribute on the **mmchconfig** command. See "GPFS configuration attributes for DMAPI" on page 579. The default is immediate timeout.

GPFS keeps the state of a failed session for 24 hours, during which the session should be assumed. When this time has elapsed, and the session has not been assumed, the session is discarded. An attempt to assume a session after it has been discarded will fail.

Event recovery

Synchronous events are recoverable after session failure.

The state of synchronous events is maintained both at the source node and at the session node. When the old session is assumed, pending synchronous events are resubmitted by surviving source nodes.

All the events originating from the session node itself are lost during session failure, including user events generated by the DM application. All file operations on the session node fail with the **ESTALE** error code.

When a session fails, all of its tokens become obsolete. After recovery, the **dm_getall_tokens** function returns an empty list of tokens, and it is therefore impossible to identify events that were outstanding when the failure occurred. All recovered events return to the initial non-received state, and must be explicitly received again. The token id of a recovered event is the same as prior to the failure (except for the mount event).

If the token of a recovered event is presented in any DMAPI function before the event is explicitly received again, the call will fail with the **ESRCH** error code. The **ESRCH** error indicates that the event exists, but is not in the outstanding state. This is to be distinguished from the **EINVAL** error code, which indicates that the token id itself is not valid (there is no event).

The semantics of the **ESRCH** error code in GPFS are different from the XDSM standard. This is entailed by the enhanced failure model. The DM application may not notice that the GPFS daemon has failed and recovered, and may attempt to use a token it has received prior to the failure. For example, it may try to respond to the event. The **ESRCH** error code tells the DM application that it must receive the event again, before it can continue using the token. Any access rights associated with the token prior to the failure are lost. See "Loss of access rights."

When a mount event is resubmitted to a session during session recovery, it will have a different token id than before the failure. This is an exception to the normal behavior, since all other recovered events have the same token id as before. The DM application thus cannot distinguish between recovered and new mount events. This should not be a problem, since the DM application must in any case be able to handle multiple mount events for the same file system.

Unmount events will not be resubmitted after session recovery. All such events are lost. This should not be a problem, since the event cannot affect the unmount operation, which has already been completed by the time the event was generated. In other words, despite being synchronous, semantically the unmount event resembles an asynchronous post event.

Loss of access rights

When the GPFS daemon fails on the session node, all file systems on the node are forced unmounted. As a result, all DM access rights associated with any local session are lost.

After daemon recovery, when the old sessions are assumed and the events are resubmitted, there is no way of identifying events that were already being handled prior to the failure (outstanding events), nor is there a guarantee that objects have not been accessed or modified after the access rights were lost. The DM application must be able to recover consistently without depending on persistent access rights. For example, it could keep its own state of events in progress, or process events idempotently.

Similarly, when a specific file system is forced unmounted at the session node, all DM access rights associated with the file system are lost, although the events themselves prevail on the session queue. After the file system is remounted, DMAPI calls using existing tokens may fail due to insufficient access rights. Also, there is no guarantee that objects have not been accessed or modified after the access rights were lost.

DODeferred deletions

The asynchronous recovery code supports deferred deletions if there are no external mounts at the time of recovery.

Once a node successfully generates a mount event for an external mount, the **sgmgr** node will start deferred deletions if it is needed. Any internal mounts would bypass deferred deletions if the file system is DMAPI enabled.

DM application failure

If only the DM application fails, the session itself remains active, events remain pending, and client threads remain blocked waiting for a response. New events will continue to arrive at the session queue.

Note: GPFS is unable to detect that the DM application has failed.

The failed DM application must be recovered on the same node, and continue handling the events. Since no DMAPI resources are lost in this case, there is little purpose in moving the DM application to another node. Assuming an existing session on another node is not permitted in GPFS, except after session node failure.

If the DM application fails simultaneously with the session node, the **gpfsready** shell script can be used to restart the DM application on the failed node. See "Initializing the Data Management application" on page 581. In the case of simultaneous failures, the DM application can also be moved to another node and assume the failed session there. See "Single-node failure" on page 602.

Chapter 3. 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 18 summarizes the GPFS programming interfaces.

Table 18. GPFS programming interfaces

Interface	Purpose
"gpfs_acl_t structure" on page 610	Contains buffer mapping for the **gpfs_getacl()** and **gpfs_putacl()** subroutines.
"gpfs_clone_copy() subroutine" on page 611	Creates a file clone of a read-only clone parent file.
"gpfs_clone_snap() subroutine" on page 613	Creates a read-only clone parent from a source file.
"gpfs_clone_split() subroutine" on page 615	Splits a file clone from its clone parent.
"gpfs_clone_unsnap() subroutine" on page 617	Changes a clone parent with no file clones back to a regular file.
"gpfs_close_inodescan() subroutine" on page 619	Closes an inode scan.
"gpfs_cmp_fssnapid() subroutine" on page 620	Compares two file system snapshot IDs.
"gpfs_declone() subroutine" on page 622	Removes file clone references to clone parent blocks.
"gpfs_direntx_t structure" on page 624	Contains attributes of a GPFS directory entry.
"gpfs_direntx64_t structure" on page 626	Contains attributes of a GPFS directory entry.
"gpfs_fcntl() subroutine" on page 628	Performs operations on an open file.
"gpfs_fgetattrs() subroutine" on page 631	Retrieves all extended file attributes in opaque format.
"gpfs_fputattrs() subroutine" on page 633	Sets all the extended file attributes for a file.
"gpfs_fputattrswithpathname() subroutine" on page 635	Sets all of the extended file attributes for a file and invokes the policy engine for **RESTORE** rules.
"gpfs_free_fssnaphandle() subroutine" on page 637	Frees a GPFS file system snapshot handle.
"gpfs_fssnap_handle_t structure" on page 638	Contains a handle for a GPFS file system or snapshot.
"gpfs_fssnap_id_t structure" on page 639	Contains a permanent identifier for a GPFS file system or snapshot.
"gpfs_fstat() subroutine" on page 640	Returns exact file status for a GPFS file.
"gpfs_get_fsname_from_fssnaphandle() subroutine" on page 642	Obtains a file system name from its snapshot handle.
"gpfs_get_fssnaphandle_by_fssnapid() subroutine" on page 643	Obtains a file system snapshot handle using its snapshot ID.
"gpfs_get_fssnaphandle_by_name() subroutine" on page 645	Obtains a file system snapshot handle using its name.
"gpfs_get_fssnaphandle_by_path() subroutine" on page 647	Obtains a file system snapshot handle using its path name.
"gpfs_get_fssnapid_from_fssnaphandle() subroutine" on page 649	Obtains a file system snapshot ID using its handle.
"gpfs_get_pathname_from_fssnaphandle() subroutine" on page 651	Obtains a file system path name using its snapshot handle.
"gpfs_get_snapdirname() subroutine" on page 653	Obtains the name of the directory containing global snapshots.

© Copyright IBM Corporation © IBM 2014, 2016

Table 18. GPFS programming interfaces (continued)

Interface	Purpose
"gpfs_get_snapname_from_fssnaphandle() subroutine" on page 655	Obtains a snapshot name using its file system snapshot handle.
"gpfs_getacl() subroutine" on page 657	Retrieves the access control information for a GPFS file.
"gpfs_iattr_t structure" on page 659	Contains attributes of a GPFS inode.
"gpfs_iattr64_t structure" on page 662	Contains attributes of a GPFS inode.
"gpfs_iclose() subroutine" on page 666	Closes a file given its inode file handle.
"gpfs_ifile_t structure" on page 668	Contains a handle for a GPFS inode.
"gpfs_igetattrs() subroutine" on page 669	Retrieves extended file attributes in opaque format.
"gpfs_igetattrsx() subroutine" on page 671	Retrieves extended file attributes; provides an option to include DMAPI attributes.
"gpfs_igetfilesetname() subroutine" on page 673	Returns the name of the fileset defined by a fileset ID.
"gpfs_igetstoragepool() subroutine" on page 675	Returns the name of the storage pool for the given storage pool ID.
"gpfs_iopen() subroutine" on page 677	Opens a file or directory by inode number.
"gpfs_iopen64() subroutine" on page 679	Opens a file or directory by inode number.
"gpfs_iputattrsx() subroutine" on page 681	Sets the extended file attributes for a file.
"gpfs_iread() subroutine" on page 684	Reads a file opened by **gpfs_iopen()**.
"gpfs_ireaddir() subroutine" on page 686	Reads the next directory entry.
"gpfs_ireaddir64() subroutine" on page 688	Reads the next directory entry.
"gpfs_ireadlink() subroutine" on page 690	Reads a symbolic link by inode number.
"gpfs_ireadlink64() subroutine" on page 692	Reads a symbolic link by inode number.
"gpfs_ireadx() subroutine" on page 694	Performs block level incremental read of a file within an incremental inode scan.
"gpfs_iscan_t structure" on page 697	Contains a handle for an inode scan of a GPFS file system or snapshot.
"gpfs_lib_init() subroutine" on page 698	Sets up a GPFS interface for additional calls.
"gpfs_lib_term() subroutine" on page 699	Cleans up after GPFS interface calls have been completed.
"gpfs_next_inode() subroutine" on page 700	Retrieves the next inode from the inode scan.
"gpfs_next_inode64() subroutine" on page 702	Retrieves the next inode from the inode scan.
"gpfs_next_inode_with_xattrs() subroutine" on page 704	Retrieves the next inode and its extended attributes from the inode scan.
"gpfs_next_inode_with_xattrs64() subroutine" on page 706	Retrieves the next inode and its extended attributes from the inode scan.
"gpfs_next_xattr() subroutine" on page 708	Returns individual attributes and their values.
"gpfs_opaque_acl_t structure" on page 710	Contains buffer mapping for the **gpfs_getacl()** and **gpfs_putacl()** subroutines.
"gpfs_open_inodescan() subroutine" on page 711	Opens an inode scan of a file system or snapshot.
"gpfs_open_inodescan64() subroutine" on page 714	Opens an inode scan of a file system or snapshot.
"gpfs_open_inodescan_with_xattrs() subroutine" on page 717	Opens an inode file and extended attributes for an inode scan.
"gpfs_open_inodescan_with_xattrs64() subroutine" on page 720	Opens an inode file and extended attributes for an inode scan.
"gpfs_prealloc() subroutine" on page 723	Pre-allocates disk storage for a GPFS file.

Table 18. GPFS programming interfaces (continued)

Interface	Purpose
"gpfs_putacl() subroutine" on page 725	Restores the access control information for a GPFS file.
"gpfs_quotactl() subroutine" on page 727	Manipulates disk quotas on file systems.
"gpfs_quotaInfo_t structure" on page 730	Contains buffer mapping for the **gpfs_quotactl()** subroutine.
"gpfs_seek_inode() subroutine" on page 732	Advances an inode scan to the specified inode number.
"gpfs_seek_inode64() subroutine" on page 734	Advances an inode scan to the specified inode number.
"gpfs_stat() subroutine" on page 736	Returns exact file status for a GPFS file.
"gpfs_stat_inode() subroutine" on page 738	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode64() subroutine" on page 740	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode_with_xattrs() subroutine" on page 742	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfs_stat_inode_with_xattrs64() subroutine" on page 744	Seeks the specified inode and retrieves that inode and its extended attributes from the inode scan.
"gpfsFcntlHeader_t structure" on page 746	Contains declaration information for the **gpfs_fcntl()** subroutine.
"gpfsGetFilesetName_t structure" on page 747	Obtains the fileset name of a file.
"gpfsGetReplication_t structure" on page 748	Obtains the replication factors of a file.
"gpfsGetSetXAttr_t structure" on page 750	Obtains or sets extended attribute values.
"gpfsGetSnapshotName_t structure" on page 752	Obtains the snapshot name of a file.
"gpfsGetStoragePool_t structure" on page 753	Obtains the storage pool name of a file.
"gpfsListXAttr_t structure" on page 754	Lists extended attributes.
"gpfsRestripeData_t structure" on page 755	Restripes the data blocks of a file.
"gpfsSetReplication_t structure" on page 757	Sets the replication factors of a file.
"gpfsSetStoragePool_t structure" on page 759	Sets the assigned storage pool of a file.

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.

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 613
Creates a read-only clone parent from a source file.

"gpfs_clone_split() subroutine" on page 615
Splits a file clone from its clone parent.

"gpfs_clone_unsnap() subroutine" on page 617
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.

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 611
Creates a file clone of a read-only clone parent file.

"gpfs_clone_split() subroutine" on page 615
Splits a file clone from its clone parent.

"gpfs_clone_unsnap() subroutine" on page 617
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 611
Creates a file clone of a read-only clone parent file.

"gpfs_clone_snap() subroutine" on page 613
Creates a read-only clone parent from a source file.

"gpfs_clone_unsnap() subroutine" on page 617
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 611
Creates a file clone of a read-only clone parent file.
"gpfs_clone_snap() subroutine" on page 613
Creates a read-only clone parent from a source file.
"gpfs_clone_split() subroutine" on page 615
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() 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_INVAL_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() 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 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() 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);
```

```
    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_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_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_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 633 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() 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: 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_fsname_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_fsname_from_fssnaphandle(gpfs_fssnap_handle_t *fssnapHandle);
```

Description

The **gpfs_get_fsname_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_fsname_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_fsname_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_fsname_from_fssnaphandle()** subroutine is not available.

EPERM
The caller does not have superuser privileges.

GPFS_E_INVAL_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_INVAL_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_E_INVAL_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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

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_INVAL_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.

ERANGE
The buffer is too small to return the snapshot directory name.

ESTALE
The cached file system information was not valid.

GPFS_E_INVAL_FSSNAPHANDLE
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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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:

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 */
```

```
#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_RPATTR         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.

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_iclose() 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_iclose(gpfs_ifile_t *ifile);
```

Description

The **gpfs_iclose()** subroutine closes an open file descriptor created by **gpfs_iopen()**.

For an overview of using **gpfs_iclose()** in a backup application, see the topic *Using APIs to develop backup applications* in the *IBM Spectrum Scale: 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_iclose()** subroutine returns void.

Exceptions

None.

Error status

Error codes include but are not limited to the following:

ENOSYS
The **gpfs_iclose()** 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_iclose()**, 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 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:

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_INVAL_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_igetattrs()** 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_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.

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_INVAL_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.

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_INVAL_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* in *IBM Spectrum Scale: 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

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.

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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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* in *IBM Spectrum Scale: 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

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.

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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_IATTR
The **iattr** structure was corrupted.

GPFS_E_INVAL_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_fputattrs()**.

The buffer passed in should contain extended attribute data that was obtained by a previous call to **gpfs_fgetattrs()**.

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_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_INVAL_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() 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* in *IBM Spectrum Scale: 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()**.

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_INVAL_IFILE
Incorrect **ifile** parameter.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

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* in *IBM Spectrum Scale: 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

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_INVAL_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() 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* in *IBM Spectrum Scale: 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

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_INVAL_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() 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_INVAL_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() 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_INVAL_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() 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* in *IBM Spectrum Scale: Administration Guide*.

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_INVAL_IFILE
Incorrect **ifile** parameter.

GPFS_E_INVAL_ISCAN
Incorrect **iscan** parameter.

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() 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() 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* in *IBM Spectrum Scale: Administration Guide*.

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.

iattr
Pointer to the returned pointer to the inode's **iattr**.

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_INVAL_FSSNAPID
The file system snapshot ID is not valid.

GPFS_E_INVAL_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() 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* in *IBM Spectrum Scale: Administration Guide*.

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_INVAL_FSSNAPID
The file system snapshot ID is not valid.

GPFS_E_INVAL_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() 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_INVAL_ISCAN
Incorrect parameters.

GPFS_E_INVAL_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() 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_INVAL_ISCAN
Incorrect parameters.

GPFS_E_INVAL_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() 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* in *IBM Spectrum Scale: Administration Guide*.

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

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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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() 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* in *IBM Spectrum Scale: Administration Guide*.

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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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* in *IBM Spectrum Scale: Administration Guide*.

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

- 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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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() 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* in *IBM Spectrum Scale: Administration Guide*.

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_INVAL_FSSNAPHANDLE
The file system snapshot handle is not valid.

GPFS_E_INVAL_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.

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_fgetattrs()** and **gpfs_fputattrs()** 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.

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.

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** parameters 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() 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_INVAL_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() 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_INVAL_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() 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. For more information, see *Exceptions to Open Group technical standards* in *IBM Spectrum Scale: 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

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() 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_INVAL_ISCAN
Incorrect parameters.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

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_INVAL_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() 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_INVAL_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_INVAL_ISCAN
Incorrect parameters.

Location

`/usr/lpp/mmfs/lib/libgpfs.a` for AIX

`/usr/lpp/mmfs/lib/libgpfs.so` for Linux

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 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 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 756.

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.

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 758.

errValue1
Returned value depending upon **errReason**.

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.

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 4. 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 19 summarizes the GPFS-specific user exits.

Table 19. GPFS user exits

User exit	Purpose
"mmsdrbackup user exit" on page 762	Performs a backup of the GPFS configuration data.
"nsddevices user exit" on page 763	Identifies local physical devices that are used as GPFS Network Shared Disks (NSDs).
"syncfsconfig user exit" on page 764	Keeps file system configuration data in replicated clusters synchronized.

© Copyright IBM Corporation © IBM 2014, 2016

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 "mmcrcluster command" on page 218.

Note: The **mmsdrbackup** user exit is supported from IBM Spectrum Scale 4.2.0 or later. It must not be used on clusters which have nodes running earlier versions of IBM Spectrum Scale.

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 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`

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.

© Copyright IBM Corporation © IBM 2014, 2016

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

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

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

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 Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

© Copyright IBM Corporation © IBM 2014, 2016

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.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

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-IBMproducts. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

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 actual people or business enterprises 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:

© (your company name) (year).
Portions of this code are derived from IBM Corp.
Sample Programs. © Copyright IBM Corp. _enter the year or years_.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

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 Export Services (CES) nodes
A subset of nodes configured within a cluster to provide a solution for exporting GPFS file systems by using the Network File System (NFS), Server Message Block (SMB), and Object protocols.

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,

© Copyright IBM Corporation © IBM 2014, 2016

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.

Microsoft Management Console (MMC)
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.

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 IBM Spectrum Protect 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.

transparent cloud tiering (TCT)
A separately installable add-on feature of IBM Spectrum Scale that provides a native cloud storage tier. It allows data center administrators to free up on-premise storage capacity, by moving out cooler data to the cloud storage, thereby reducing capital and operational expenditures. .

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 518
--trace-dispatch
  changing 518
--tracedev-buffer-size
  changing 518
--tracedev-compression-level
  changing 518
--tracedev-overwrite-buffer-size
  changing 519
--tracedev-write-mode
  changing 519
--traceFileSize
  changing 518

A

access control information 657
  restoring 725
access control lists
  creating 450
  deleting 263
  displaying 317
  editing 295
  getting 320
access rights
  locking 575
  loss of 604
  restrictions 575
accessibility features for IBM Spectrum Scale 765
ACL information 610, 710
  restoring 725
  retrieving 657
AD server
  querying 32
adding
  disks 23
adding nodes to a GPFS cluster 29
additional calls, setup of interface for 698
adminMode attribute 126
AFM 37, 40, 54, 447
appendOnly file attribute 113, 352
application failure 604
argument
  buflen 601
  flags 601
  hanp 599
  hlen 599
  len 601
  nelem 599, 601
  nelemp 599
  off 601
  sessinfop 599
atime 171, 711, 714, 717, 720, 736, 738, 740
atimeDeferredSeconds attribute 128
attribute bit
  dm_at_size 584
attributes
  adminMode 126
attributes *(continued)*
  atimeDeferredSeconds 128
  autoload 128
  automountDir 129
  cesSharedRoot 129
  cipherList 129
  cnfsGrace 129
  cnfsMountdPort 129
  cnfsNFSDprocs 130
  cnfsReboot 130
  cnfsSharedRoot 130
  cnfsVersions 130
  commandAudit 130
  configuration 570
  dataDiskWaitTimeForRecovery 130
  dataStructureDump 131
  deadlockBreakupDelay 131
  deadlockDataCollectionDailyLimit 131
  deadlockDataCollectionMinInterval 131
  deadlockDetectionThreshold 131
  deadlockDetectionThresholdForShortWaiters 131
  deadlockOverloadThreshold 131
  debugDataControl 131
  defaultHelperNodes 132
  defaultMountDir 132
  description 577
  disableInodeUpdateOnFdatasync 132
  dmapiDataEventRetry 132
  dmapiEventTimeout 132
  dmapiMountEvent 133
  dmapiMountTimeout 133
  dmapiSessionFailureTimeout 133
  enableIPv6 134
  enforceFilesetQuotaOnRoot 134
  expelDataCollectionDailyLimit 134
  expelDataCollectionMinInterval 134
  extended 577
  failureDetectionTime 134
  fastestPolicyCmpThreshold 134
  fastestPolicyMaxValidPeriod 134
  fastestPolicyMinDiffPercent 135
  fastestPolicyNumReadSamples 135
  fileHeatLossPercent 135
  fileHeatPeriodMinutes 135
  FIPS1402mode 135
  forceLogWriteOnFdatasync 135
  GPFS-specific 582
  lrocData 135
  lrocDataMaxFileSize 136
  lrocDataStubFileSize 136
  lrocDirectories 136
  lrocInodes 136
  maxblocksize 136
  maxDownDisksForRecovery 137
  maxFailedNodesForRecovery 137
  maxFcntlRangesPerFile 137
  maxFilesToCache 137
  maxMBpS 137
  maxStatCache 137
  metadataDiskWaitTimeForRecovery 137
  minDiskWaitTimeForRecovery 138

© Copyright IBM Corp. 2014, 2016

attributes *(continued)*
mmapRangeLock 138
nistCompliance 138
non-opaque 571, 577
noSpaceEventInterval 139
nsdBufSpace 139
nsdRAIDBufferPoolSizePct 140
nsdRAIDTracks 140
nsdServerWaitTimeForMount 140
nsdServerWaitTimeWindowOnMount 140
numaMemoryInterleave 141
opaque 571, 577
pagepool 141
pagepoolMaxPhysMemPct 141
prefetchThreads 141
readReplicaPolicy 142
release 142
restripeOnDiskFailure 143
rpcPerfNumberDayIntervals 143
rpcPerfNumberHourIntervals 143
rpcPerfNumberMinuteIntervals 143
rpcPerfNumberSecondIntervals 143
rpcPerfRawExecBufferSize 143
rpcPerfRawStatBufferSize 144
sidAutoMapRangeLength 144
sidAutoMapRangeStart 144
subnets 144
systemLogLevel 145
tiebreakerDisks 145
uidDomain 146
unmountOnDiskFail 146
usePersistentReserve 146
verbsPorts 147
verbsRdma 147
verbsRdmaCm 147
verbsRdmaRoCEToS 147
verbsRdmaSend 148
verbsRdmasPerConnection 148
verbsRdmasPerNode 148
verbsSendBufferMemoryMB 148
worker1Threads 148
autoload attribute 128
automated installation toolkit 548
automatic mount, indicating 233
automountDir attribute 129

B

backing up a file system
configuration information 81
backup server 72
BigInsights Hadoop distribution
mmhadoopctl 323
block level incremental backups 694
block size
choosing 233
effect on maximum mounted file system size 136, 233

C

callbacks 10
daRebuildFailed 17
nsdCksumMismatch 19
pdFailed 19
pdPathDown 19
pdRecovered 19
callbacks *(continued)*
pdReplacePdisk 19
postRGRelinquish 20
postRGTakeover 20
preRGRelinquish 19
preRGTakeover 20
rgOpenFailed 20
rgPanic 21
ces
config 500
mmsmb 500
CES
configuration 96, 402
mmces 96
mmcesdr 105
mmnfs 402
mmobj 414
mmprotocoltrace 443
mmuserauth 527
protocol tracing 443
topic 527
cesSharedRoot attribute 129
changing
an administration or daemon interface for a node 180
attributes
adminMode 126
atimeDeferredSeconds 128
autoload 128
automountDir 129
cesSharedRoot 129
cipherList 129
cluster configuration 123
cnfsGrace 129
cnfsMountdPort 129
cnfsNFSDprocs 130
cnfsReboot 130
cnfsSharedRoot 130
cnfsVersions 130
dataDiskWaitTimeForRecovery 130
dataStructureDump 131
deadlockBreakupDelay 131
deadlockDataCollectionDailyLimit 131
deadlockDataCollectionMinInterval 131
deadlockDetectionThreshold 131
deadlockDetectionThresholdForShortWaiters 131
deadlockOverloadThreshold 131
debugDataControl 131
defaultHelperNodes 132
defaultMountDir 132
disableInodeUpdateOnFdatasync 132
dmapiDataEventRetry 132
dmapiEventTimeout 132
dmapiMountEvent 133
dmapiMountTimeout 133
dmapiSessionFailureTimeout 133
enableIPv6 134
enforceFilesetQuotaOnRoot 134
expelDataCollectionDailyLimit 134
expelDataCollectionMinInterval 134
failureDetectionTime 134
fastestPolicyCmpThreshold 134
fastestPolicyMaxValidPeriod 134
fastestPolicyMinDiffPercent 135
fastestPolicyNumReadSamples 135
fileHeatLossPercent 135
fileHeatPeriodMinutes 135
FIPS1402mode 135

changing *(continued)*
attributes *(continued)*
forceLogWriteOnFdatasync 135
lrocData 135
lrocDataMaxFileSize 136
lrocDataStubFileSize 136
lrocDirectories 136
lrocInodes 136
maxblocksize 136
maxDownDisksForRecovery 137
maxFailedNodesForRecovery 137
maxFcntlRangesPerFile 137
maxFilesToCache 137
maxMBpS 137
maxStatCache 137
metadataDiskWaitTimeForRecovery 137
minDiskWaitTimeForRecovery 138
mmapRangeLock 138
nistCompliance 138
noSpaceEventInterval 139
nsdBufSpace 139
nsdRAIDBufferPoolSizePct 140
nsdRAIDTracks 140
nsdServerWaitTimeForMount 140
nsdServerWaitTimeWindowOnMount 140
numaMemoryInterleave 141
pagepool 141
pagepoolMaxPhysMemPct 141
prefetchThreads 141
readReplicaPolicy 142
release 142
restripeOnDiskFailure 143
rpcPerfNumberDayIntervals 143
rpcPerfNumberHourIntervals 143
rpcPerfNumberMinuteIntervals 143
rpcPerfNumberSecondIntervals 143
rpcPerfRawExecBufferSize 143
rpcPerfRawStatBufferSize 144
sidAutoMapRangeLength 144
sidAutoMapRangeStart 144
subnets 144
systemLogLevel 145
tiebreakerDisks 145
uidDomain 146
unmountOnDiskFail 146
usePersistentReserve 146
verbsPorts 147
verbsRdma 147
verbsRdmaCm 147
verbsRdmaRoCEToS 147
verbsRdmaSend 148
verbsRdmasPerConnection 148
verbsRdmasPerNode 148
verbsSendBufferMemoryMB 148
worker1Threads 148
disk parameters 151
disk states 151
fileset attributes 163
tracing attributes 517
user-defined node classes 185
changing Quality of Service for I/O operations (QoS) level 196
changing storage pool properties 194
cipherList attribute 68, 129
cleanup after GPFS interface calls 699
Client license 175
client node
refresh NSD server 412
clone, file
copy 201, 611
decloning 622
redirect 201
show 201
snap 201, 613
split 201, 615
unsnap 617
cluster
changing configuration attributes 123
changing tracing attributes 517
configuration data 302
cluster configuration attributes
displaying 359
cluster configuration data 314
cluster configuration server 218, 313
Cluster Export Services
config 500
configuration 96, 402
mmces 96
mmcesdr 105
mmnfs 402
mmobj 414
mmprotocoltrace 443
mmsmb 500
mmuserauth 527
protocol tracing 443
topic 527
cnfsGrace attribute 129
cnfsMountdPort attribute 129
cnfsNFSDprocs attribute 130
cnfsReboot attribute 130
cnfsSharedRoot attribute 130
cnfsVersions attribute 130
commandAudit attribute 130
commands 1
gpfs.snap 5
mmaddcallback 10
mmadddisk 23, 764
mmaddnode 29
mmadquery 32
mmafmconfig 37
mmafmctl 40
mmafmlocal 54
mmapplypolicy 56
mmauth 67
mmbackup 72
mmbackupconfig 81
mmbuildgpl 83
mmcallhome 85
mmces 96
mmcesdr 105
mmchattr 113
mmchcluster 119
mmchconfig 123, 579
mmchdisk 151
mmcheckquota 159
mmchfileset 163
mmchfs 169, 580
mmchlicense 175
mmchmgr 178
mmchnode 180
mmchnodeclass 185
mmchnsd 188
mmchpolicy 191

commands *(continued)*
mmchpool 194
mmchqos 196
mmclone 201
mmcloudgateway 204
mmcrcluster 218
mmcrfileset 223
mmcrfs 229, 580
mmcrnodeclass 239
mmcrnsd 23, 241, 485, 763, 764
mmcrsnapshot 246
mmdefedquota 251
mmdefquotaoff 254
mmdefquotaon 257
mmdefragfs 260
mmdelacl 263
mmdelcallback 265
mmdeldisk 266, 764
mmdelfileset 271
mmdelfs 274
mmdelnode 276
mmdelnodeclass 279
mmdelnsd 281
mmdelsnapshot 283
mmdf 287
mmdiag 290
mmeditacl 295
mmedquota 298
mmexportfs 302
mmfsck 304
mmfsctl 313
mmgetacl 317
mmgetstate 320
mmhadoopctl 323
mmhealth 324, 340
mmimgbackup 329
mmimgrestore 333
mmimportfs 336
mmlinkfileset 350
mmlsattr 352
mmlscallback 355
mmlscluster 357
mmlsconfig 359
mmlsdisk 361
mmlsfileset 365
mmlsfs 369
mmlslicense 373
mmlsmgr 375
mmlsmount 377
mmlsnodeclass 379
mmlsnsd 381
mmlspolicy 384
mmlspool 386
mmlsqos 388
mmlsquota 391
mmlssnapshot 395, 661, 665
mmmigratefs 398
fastea 398
mmmount 400
mmnfs 402
mmnsddiscover 412
mmobj 414
mmperfmon query 428
mmpmon 438
mmprotocoltrace 443
mmpsnap 447
mmputacl 450
commands *(continued)*
mmquotaoff 453
mmquotaon 455
mmremotecluster 457
mmremotefs 460
mmrepquota 463
mmrestoreconfig 467
mmrestorefs 471
mmrestripefile 475
mmrestripefs 478
mmrpldisk 485
mmsdrrestore 492
mmsetquota 494
mmshutdown 498
mmsmb 500
mmsnapdir 511, 638
mmstartup 515
mmtracectl 517
mmumount 521
mmunlinkfileset 524
mmuserauth 527
mmwinservctl 546
spectrumscale 548
configuration attributes
DMAPI 579
dmapiEnable 580
dmapiEventTimeout 577
NFS (Network File System) 579
dmapiMountTimeout 575, 580
dmapiSessionFailureTimeout 580, 603
connector for Hadoop distributions, GPFS
mmhadoopctl 323
consistency checks 32
contact node 277
creating
access control lists 450
file systems 229
filesets 223
ctime 711, 714, 717, 720

D

daRebuildFailed callback 17
Data Management API
failure 604
restarting 604
data replica 171
data structures
defined 581
specific to GPFS implementation 581
data type
dm_eventset_t 582
dataDiskWaitTimeForRecovery attribute 130
dataStructureDump attribute 131
deadlockBreakupDelay attribute 131
deadlockDataCollectionDailyLimit attribute 131
deadlockDataCollectionMinInterval attribute 131
deadlockDetectionThreshold attribute 131
deadlockDetectionThresholdForShortWaiters attribute 131
deadlockOverloadThreshold attribute 131
debugDataControl attribute 131
declone 622
default quotas
activating 257
deactivating 254
editing 251
defaultHelperNodes attribute 132

defaultMountDir attribute 132
definitions
  GPFS-specific DMAPI functions 584, 586, 587, 589, 591, 593, 595, 597
deleting
  disks 266
  file systems 274
  filesets 271
  nodes from a cluster 276
  snapshots 283
deleting links
  snapshots 511
deleting, Network Shared Disks (NSDs) 281
deny-write open lock 169
description
  dmapiDataEventRetry 579
  dmapiFileHandleSize 579
  dmapiMountEvent 579
directives
  subroutine for passing 628
directory
  /usr/lpp/mmfs/bin 579
  /usr/lpp/mmfs/include 578
  /usr/lpp/mmfs/lib 578
  /usr/lpp/mmfs/samples 581
  /var/mmfs/etc 581
directory entry
  reading 686, 688
disableInodeUpdateOnFdatasync attribute 132
disaster recovery 313
disk access
  path discovery 412
disk descriptor 152
disk parameter
  changing 151
disk state
  changing 151
  suspended 151
disk storage
  pre-allocating 723
disk usage 151, 485
disks
  adding 23, 485
  configuration 361
  deleting 266
  displaying state 361
  reducing fragmentation 260
  replacing 485
displaying
  access control lists 317
  cluster configuration attributes 359
  disk state 361
  filesets 365
  GPFS cluster configuration information 357
  NSD belonging to a GPFS cluster 381
  quotas 391
  snapshots 395
displaying Quality of Service for I/O operations (QoS) settings 388
DM application threads 576
DM application, role in session failure 572
DM_EVENT_POSTPERMCHANGE 600
DM_EVENT_PREPERMCHANGE 600
dm_handle_to_snap
  definitions 586
dm_make_xhandle
  definitions 587
DM_NO_TOKEN 575
dm_remove_dmattr_nosync
  definitions 589
dm_set_dmattr_nosync
  definitions 591
dm_set_eventlist_nosync
  definitions 593
dm_set_region_nosync
  definitions 595
dm_sync_dmattr_by_handle
  definitions 597
DMAPI 172
  administration 578
  applications 578
  compiling on AIX nodes 579
  configuration attributes 570, 579
  failure 601, 604
  features 565
  files on Linux nodes 579
  functions 566
  initializing 581
  overview 565
  recovery 601
  restarting 604
  restrictions 571
DMAPI events
  GPFS-specific 565
  GPFS-specific attribute events that are not part of the DMAPI standard 566
  implemented in DMAPI for GPFS 565
  optional events not implemented in DMAPI for GPFS 566
DMAPI events, GPFS-specific 599
DMAPI functions
  error code
    EIO 600
    ENOMEM 600
    ENOSYS 600
    ENOTREADY 600
    EPERM 600
    ESTALE 600
DMAPI functions, GPFS-specific 570
  definitions 584
  dm_handle_to_snap 586
  dm_make_xhandle 587
  dm_remove_dmattr_nosync 589
  dm_set_dmattr_nosync 591
  dm_set_eventlist_nosync 593
  dm_set_region_nosync 595
  dm_sync_dmattr_by_handle 597
DMAPI token, description 575
dmapiDataEventRetry
  description 579
dmapiDataEventRetry attribute 132
dmapiEventTimeout attribute 132
dmapiFileHandleSize
  description 579
dmapiMountEvent attribute 133
  description 579
dmapiMountTimeout attribute 133
dmapiSessionFailureTimeout attribute 133
DODeferred deletions 604
dumps, storage of information 131

E

editing
  default quotas 251

enableIPv6 attribute 134
enabling DMAPI
  migrating a file system 580
  mmchfs command 580
  mmcrfs command 580
enforceFilesetQuotaOnRoot attribute 134
environment
  multiple-node 572, 601
  single-node 572, 601
error code
  EAGAIN 583
  EBADF 583, 584, 600
  EBUSY 575, 577
  EINVAL 584, 600, 604
  EIO 574, 581
  ENOSYS 583
  ENOTREADY 577, 583, 603
  EPERM 583, 600
  ESRCH 584, 600, 603, 604
error code, definitions 600
events
  as defined in XDSM standard 565
  asynchronous 566, 573
  description 572
  disposition 572
  enabled 573
  GPFS-specific attribute events that are not part of the DMAPI standard 566
  GPFS-specific DMAPI events 565, 599
  implemented
    data events 566
    file system administration 565
    metadata events 566
    namespace events 566
    pseudo events 566
  implemented in DMAPI for GPFS 565
  mount 574
  not implemented
    file system administration 566
    metadata 566
  optional events not implemented in DMAPI for GPFS 566
  pre-unmount 574
  preunmount 582
  reliable DMAPI destroy 574
  source node 601
  synchronous 573
  unmount 574, 582
events, metadata
  DM_EVENT_POSTPERMCHANGE 600
  DM_EVENT_PREPERMCHANGE 600
expelDataCollectionDailyLimit attribute 134
expelDataCollectionMinInterval attribute 134
extended ACLs
  retrieve 631
  set 633, 635, 681
extended attributes 352
extended file attributes 633, 635, 681
  retrieve 631
  set 633, 635, 681

F

failure
  dm application 601
  GPFS daemon 566, 572
  partial system 602
  session 572, 573
failure *(continued)*
  session node 602
  single-node 602
  source node 602
  total system 602
failure group 151, 485
failureDetectionTime attribute 134
fastestPolicyCmpThreshold attribute 134
fastestPolicyMaxValidPeriod attribute 134
fastestPolicyMinDiffPercent attribute 135
fastestPolicyNumReadSamples attribute 135
field
  dt_change 582
  dt_ctime 582
  dt_dtime 582
  dt_nevents 599
  ev_nodeid 582
  me_handle2 575
  me_mode 575, 582, 599
  me_name1 575
  me_roothandle 575
  ne_mode 582
  rg_opaque 582
  uio_resid 601
file
  /etc/filesystems 575
  access control information 653, 657, 725
  ACL information 653, 657, 725
  block level incremental read 694
  dmapi_types.h 578
  dmapi.exp export 578
  dmapi.h 578
  dmapicalls 579, 583
  extended attributes 631, 633, 635, 681
  reading 684
file access pattern information 628
file attribute
  extended 669, 671
  querying 352
file attributes
  appendOnly 113, 352
file clone
  copy 201, 611
  decloning 622
  redirect 201
  show 201
  snap 201, 613
  split 201, 615
  unsnap 617
file descriptor
  closing 666
  opening 677, 679
file handle
  error code 600
file status information 640, 736, 738, 740
  gpfs_stat_inode_with_xattrs() 742
  gpfs_stat_inode_with_xattrs64() 744
file system descriptor quorum 314
file system handle 575
  usage of 598
file system manager
  changing nodes 178
  displaying current 375
file system name 642, 645
file system snapshot handle 638
file system space
  querying 287

file systems
adding disks 23
backing up 72
block size 229
change manager node 178
changing attributes 169
changing attributes for files 113
checking 304, 336
control request 313
creating 229
creating snapshot 246
deleting 274
deleting disks 266
displaying attributes 369
displaying format version 369
exporting 302
file system manager
displaying 375
format version 169
formatting 230
handle 638
importing 336
inconsistencies 304
links to snapshots 511
listing mounted 377
migrating 169, 398
mounted file system sizes 136, 233
mounting 400
moving to another cluster 169, 302
mtime value 170
querying space 287
quotas 257
rebalancing 478
reducing fragmentation 260
remote 67, 457, 460
repairing 304
restoring configuration information 467
restoring with snapshots 471
restripe 27
restriping 478
unmounting 498, 521
fileHeatLossPercent attribute 135
fileHeatPeriodMinutes attribute 135
files
orphaned 304
rebalancing 475
restriping 475
files, memory mapped 577
files, required 578
fileset quota 252, 463
filesets
changing attributes 163
creating 223
deleting 271
displaying 365
ID 673
linking 350
name 673
restoring with snapshots 471
unlinking 524
FIPS1402mode attribute 135
flag
DM_LOCAL_MOUNT 574, 575, 582
DM_MOUNT_LOCAL 599
DM_MOUNT_REMOTE 599
DM_REMOTE_MOUNT 575, 582
DM_RR_WAIT 583
flag *(continued)*
DM_UNMOUNT_FORCE 575
FlashCopy image 313
forceLogWriteOnFdatasync attribute 135
FPO license 175
full backup 700, 702, 711, 714, 717, 720
function
dm_create_session 599
dm_downgrade_right 583, 601
dm_find_eventmsg 601
dm_get_alloc_info 601
dm_get_bulkall 583, 584, 598, 599, 601
dm_get_bulkattr 583, 584, 598, 601
dm_get_config 570
dm_get_config_events 570, 599
dm_get_dirattrs 601
dm_get_eventlist 583, 584, 598, 599
dm_get_events 601
dm_get_fileattr 582, 599
dm_get_mount_info 584
dm_get_mountinfo 575, 582, 583, 598, 599, 601
dm_get_region 601
dm_getall_disp 598, 601
dm_getall_dmattr 601
dm_getall_sessions 601
dm_getall_tokens 601, 603
dm_handle_hash 599
dm_handle_is_valid 599
dm_handle_to_fshandle 599
dm_handle_to_fsid 599
dm_handle_to_igen 599
dm_handle_to_ino 599
dm_handle_to_path 601
dm_handle_to_snap 599
dm_init_attrloc 599
dm_init_service 601
dm_make_fshandle 598
dm_make_handle 598
dm_make_xhandle 598
dm_mount_event 575
dm_move_event 584, 601
dm_probe_hole 598, 601
dm_punch_hole 577, 584, 598, 601
dm_query_right 583
dm_query_session 598, 599
dm_read_invis 598, 601
dm_release_right 583
dm_request_right 583
dm_respond_event 601
dm_send_msg 598
dm_set_disp 583, 584, 598, 599
dm_set_eventlist 583, 584, 598, 599
dm_set_file_attr 584
dm_set_return_on_destroy 583, 584, 598
dm_sync_by_handle 601
dm_upgrade_right 583, 601
dm_write_invis 577, 598, 601
functions
implemented 567, 569
mandatory 567
not implemented 570
optional 569, 570
restrictions 583
functions, GPFS-specific DMAPI 570
definitions 584
dm_handle_to_snap 586
dm_make_xhandle 587

functions, GPFS-specific DMAPI *(continued)*
dm_remove_dmattr_nosync 589
dm_set_dmattr_nosync 591
dm_set_eventlist_nosync 593
dm_set_region_nosync 595
dm_sync_dmattr_by_handle 597

G

gathering data to solve GPFS problems 5
genkey 68
GPFS
access rights
loss of 604
Data Management API 565
DM application failure 604
DMAPI 565
failure 601
recovery 601
enhancements 581
failure
single-node 602
file system 565
implementation 565, 581
installation toolkit 548
license designation 175, 373
licensing 175, 373
mmhealth command 324
mmprotocoltrace command 443
programming interfaces 619, 620, 622, 624, 626, 628, 631, 633, 635, 637, 638, 639, 640, 642, 643, 645, 647, 649, 651, 653, 655, 657, 659, 662, 666, 668, 669, 671, 673, 675, 677, 679, 681, 684, 686, 688, 690, 692, 694, 697, 698, 699, 700, 702, 704, 706, 708, 710, 711, 714, 717, 720, 723, 725, 727, 730, 732, 734, 736, 738, 740, 742, 744, 746, 747, 748, 750, 752, 753, 754, 755, 757, 759
session
failure 603
recovery 603
stopping 498
GPFS cluster
creating 218
GPFS cluster configuration data 302
GPFS cluster configuration information
displaying 357
GPFS cluster configuration server
changing 119
primary 120
secondary 120
GPFS cluster configuration servers
choosing 218
GPFS cluster data 241
GPFS commands 1
GPFS configuration data 762, 764
GPFS connector for Hadoop distributions
mmhadoopctl 323
GPFS daemon
starting 515
stopping 498
GPFS daemon failure 572
GPFS daemon status 320
GPFS directory entry 624, 626
GPFS enhancements
implementation of 581
GPFS file system snapshot handle 642, 643, 645, 647, 649, 651, 655
free 637
GPFS portability layer 83
GPFS programming interfaces 607
GPFS subroutines 607
GPFS user exits 761
gpfs_acl_t 610
GPFS_ATTRFLAG_DEFAULT
gpfs_fgetattrs() 631
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
GPFS_ATTRFLAG_FINALIZE_ATTRS
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_IGNORE_PLACEMENT
gpfs_igetattrsx() 671
GPFS_ATTRFLAG_IGNORE_POOL
GPFS_ATTRFLAG_FINALIZE_ATTRS
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_INCL_DMAPI
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_INCL_ENCR
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_MODIFY_CLONEPARENT
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_SKIP_CLONE
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_SKIP_IMMUTABLE
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_USE_POLICY
gpfs_fgetattrs() 631
gpfs_fgetattrs() 631
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_INCL_DMAPI
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_INCL_ENCR
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_MODIFY_CLONEPARENT
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_NO_PLACEMENT
gpfs_fgetattrs() 631
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_SKIP_CLONE
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_SKIP_IMMUTABLE
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681

GPFS_ATTRFLAG_USE_POLICY
gpfs_fputattrs() 633
gpfs_fputattrswithpathname() 635
gpfs_igetattrsx() 671
gpfs_iputattrsx() 681
gpfs_clone_copy() 611
gpfs_clone_snap() 613
gpfs_clone_split() 615
gpfs_clone_unsnap() 617
gpfs_close_inodescan() 619
gpfs_cmp_fssnapid() 620
gpfs_declone() 622
gpfs_direntx_t 624
gpfs_direntx64_t 626
gpfs_fcntl() 628
gpfs_fgetattrs() 631
GPFS_ATTRFLAG_DEFAULT 631
GPFS_ATTRFLAG_FINALIZE_ATTRS 631
GPFS_ATTRFLAG_IGNORE_POOL 631
GPFS_ATTRFLAG_INCL_DMAPI 631
GPFS_ATTRFLAG_INCL_ENCR 631
GPFS_ATTRFLAG_MODIFY_CLONEPARENT 631
GPFS_ATTRFLAG_NO_PLACEMENT 631
GPFS_ATTRFLAG_SKIP_CLONE 631
GPFS_ATTRFLAG_SKIP_IMMUTABLE 631
GPFS_ATTRFLAG_USE_POLICY 631
gpfs_fputattrs() 633
GPFS_ATTRFLAG_DEFAULT 633
GPFS_ATTRFLAG_FINALIZE_ATTRS 633
GPFS_ATTRFLAG_IGNORE_POOL 633
GPFS_ATTRFLAG_INCL_DMAPI 633
GPFS_ATTRFLAG_INCL_ENCR 633
GPFS_ATTRFLAG_MODIFY_CLONEPARENT 633
GPFS_ATTRFLAG_NO_PLACEMENT 633
GPFS_ATTRFLAG_SKIP_CLONE 633
GPFS_ATTRFLAG_SKIP_IMMUTABLE 633
GPFS_ATTRFLAG_USE_POLICY 633
gpfs_fputattrswithpathname() 635
GPFS_ATTRFLAG_DEFAULT 635
GPFS_ATTRFLAG_FINALIZE_ATTRS 635
GPFS_ATTRFLAG_IGNORE_POOL 635
GPFS_ATTRFLAG_INCL_DMAPI 635
GPFS_ATTRFLAG_INCL_ENCR 635
GPFS_ATTRFLAG_MODIFY_CLONEPARENT 635
GPFS_ATTRFLAG_NO_PLACEMENT 635
GPFS_ATTRFLAG_SKIP_CLONE 635
GPFS_ATTRFLAG_SKIP_IMMUTABLE 635
GPFS_ATTRFLAG_USE_POLICY 635
gpfs_free_fssnaphandle() 637
gpfs_fssnap_handle_t 638
gpfs_fssnap_id_t 639
gpfs_fstat() 640
gpfs_get_fsname_from_fssnaphandle() 642
gpfs_get_fssnaphandle_by_fssnapid() 643
gpfs_get_fssnaphandle_by_name() 645
gpfs_get_fssnaphandle_by_path() 647
gpfs_get_fssnapid_from_fssnaphandle() 649
gpfs_get_pathname_from_fssnaphandle() 651
gpfs_get_snapdirname() 653
gpfs_get_snapname_from_fssnaphandle() 655
gpfs_getacl() 657
gpfs_iattr_t 659
gpfs_iattr64_t 662
gpfs_iclose() 666
gpfs_ifile_t 668
gpfs_igetattrs() 669
gpfs_igetattrsx() 671
gpfs_igetattrsx() *(continued)*
GPFS_ATTRFLAG_FINALIZE_ATTRS 671
GPFS_ATTRFLAG_IGNORE_PLACEMENT 671
GPFS_ATTRFLAG_INCL_DMAPI 671
GPFS_ATTRFLAG_INCL_ENCR 671
GPFS_ATTRFLAG_MODIFY_CLONEPARENT 671
GPFS_ATTRFLAG_NO_PLACEMENT 671
GPFS_ATTRFLAG_SKIP_CLONE 671
GPFS_ATTRFLAG_SKIP_IMMUTABLE 671
GPFS_ATTRFLAG_USE_POLICY 671
gpfs_igetfilesetname() 673
gpfs_igetstoragepool() 675
gpfs_iopen() 677
gpfs_iopen64() 679
gpfs_iputattrsx() 681
GPFS_ATTRFLAG_FINALIZE_ATTRS 681
GPFS_ATTRFLAG_IGNORE_POOL 681
GPFS_ATTRFLAG_INCL_DMAPI 681
GPFS_ATTRFLAG_INCL_ENCR 681
GPFS_ATTRFLAG_MODIFY_CLONEPARENT 681
GPFS_ATTRFLAG_NO_PLACEMENT 681
GPFS_ATTRFLAG_SKIP_CLONE 681
GPFS_ATTRFLAG_SKIP_IMMUTABLE 681
GPFS_ATTRFLAG_USE_POLICY 681
gpfs_iread() 684
gpfs_ireaddir() 686
gpfs_ireaddir64() 688
gpfs_ireadlink() 690
gpfs_ireadlink64() 692
gpfs_ireadx() 694
gpfs_iscan_t 697
gpfs_lib_init() 698
gpfs_lib_term() 699
gpfs_next_inode_with_xattrs() 704
gpfs_next_inode_with_xattrs64() 706
gpfs_next_inode() 700
gpfs_next_inode64() 702
gpfs_next_xattr() 708
gpfs_opaque_acl_t 710
gpfs_open_inodescan_with_xattrs() 717
gpfs_open_inodescan_with_xattrs64() 720
gpfs_open_inodescan() 711
gpfs_open_inodescan64() 714
gpfs_prealloc() 723
gpfs_putacl() 725
gpfs_quotactl() 727
gpfs_quotaInfo_t 730
gpfs_seek_inode() 732
gpfs_seek_inode64() 734
gpfs_stat_inode_with_xattrs() 742
gpfs_stat_inode_with_xattrs64() 744
gpfs_stat_inode() 738
gpfs_stat_inode64() 740
gpfs_stat() 736
GPFS-specific DMAPI events 565, 599
GPFS-specific DMAPI functions 570
definitions 584
dm_handle_to_snap 586
dm_make_xhandle 587
dm_remove_dmattr_nosync 589
dm_set_dmattr_nosync 591
dm_set_eventlist_nosync 593
dm_set_region_nosync 595
dm_sync_dmattr_by_handle 597
gpfs.snap command 5
gpfsFcntlHeader_t 746
gpfsGetFilesetName_t 747

gpfsGetReplication_t 748
gpfsGetSetXAttr_t 750
gpfsGetSnapshotName_t 752
gpfsGetStoragePool_t 753
gpfsListXAttr_t 754
gpfsRestripeData_t 755
gpfsSetReplication_t 757
gpfsSetStoragePool_t 759
grace period
  changing 298, 494
  setting 298, 494
group quota 252, 254, 257, 299, 391, 455, 463

H

Hadoop distributions, GPFS connector for
  mmhadoopctl 323
hints
  subroutine for passing 628
hole 694

I

I/O caching policy
  changing 113
IBM Spectrum Protect
  using the mmbackup command 72
IBM Spectrum Scale 329, 333, 565, 566, 567, 569, 570, 571, 572, 574, 575, 576, 577, 578, 579, 580, 581, 583, 584, 586, 587, 589, 591, 593, 595, 597, 599, 600
  access rights
    loss of 604
  commands
    mmcloudgateway 204
  Data Management API 565
  DM application failure 604
  DMAPI 565, 572
    failure 601
    recovery 601
  DMAPI functions 598
  DODeferred deletions 604
  failure
    single-node 602
  installation toolkit 548
  license designation 175, 373
  licensing 175, 373
  programming interfaces 619, 620, 622, 624, 626, 628, 631, 633, 635, 637, 638, 639, 640, 642, 643, 645, 647, 649, 651, 653, 655, 657, 659, 662, 666, 668, 669, 671, 673, 675, 677, 679, 681, 684, 686, 688, 690, 692, 694, 697, 698, 699, 700, 702, 704, 706, 708, 710, 711, 714, 717, 720, 723, 725, 727, 730, 732, 734, 736, 738, 740, 742, 744, 746, 747, 748, 750, 752, 753, 754, 755, 757, 759
  recovery
    synchronous event 603
  session
    failure 603
    recovery 603
IBM Spectrum Scale information units ix
IBM Spectrum Scale user exits 761
in-doubt value 159, 392, 464
incremental backup 700, 702, 711, 714, 717, 720
inode
  attributes 659, 662
inode file handle 666, 668
inode number 677, 679, 690, 692, 700, 702, 704, 706, 708, 711, 714, 717, 720, 732, 734
inode scan 694, 700, 702, 704, 706, 708, 732, 734
  closing 619
  opening 711, 714, 717, 720
inode scan handle 619, 697
installation 548
installation requirements 578
interface calls, cleanup after 699
interface for additional calls, setup of 698
iscan handle 619

K

kernel memory 113

L

license 175
linking
  filesets 350
links to snapshots
  creating 511
  deleting 511
listing
  snapshots 395
  user-defined callbacks 355
listing Quality of Service for I/O operations (QoS) settings 388
lrocData attribute 135
lrocDataMaxFileSize attribute 136
lrocDataStubFileSize attribute 136
lrocDirectories attribute 136
lrocInodes attribute 136

M

macro
  DM_TOKEN_EQ (x,y) 582
  DM_TOKEN_GE (x,y) 582
  DM_TOKEN_GT (x,y) 582
  DM_TOKEN_LE (x,y) 582
  DM_TOKEN_LT (x,y) 582
  DM_TOKEN_NE (x,y) 582
  DMEV_ADD(eset1, eset2) 582
  DMEV_ALL(eset) 582
  DMEV_ISALL(eset) 582
  DMEV_ISDISJ(eset1, eset2) 582
  DMEV_ISEQ(eset1, eset2) 582
  DMEV_ISSUB(eset2) 582
  DMEV_ISZERO(eset) 582
  DMEV_NORM(eset) 582
  DMEV_REM(eset1, eset2) 582
  DMEV_RES(eset1, eset2) 582
macros, GPFS 582
macros, XDSM standard 582
maxblocksize attribute 136
maxDownDisksForRecovery attribute 137
maxFailedNodesForRecovery attribute 137
maxFcntlRangesPerFile attribute 137
maxFilesToCache attribute 137
maximum number of files
  changing 169
  displaying 369
maxMBpS attribute 137
maxStatCache attribute 137

memory mapped files 577
metadata 114
metadata events
  DM_EVENT_POSTPERMCHANGE 600
  DM_EVENT_PREPERMCHANGE 600
metadata replica 170
metadataDiskWaitTimeForRecovery attribute 137
minDiskWaitTimeForRecovery attribute 138
mmaddcallback 10
mmadddisk 23, 764
mmaddnode 29
mmadquery 32
mmafmconfig 37
mmafmctl 40
mmafmlocal 54
mmapplypolicy 56
mmapRangeLock attribute 138
mmauth 67
mmbackup 72
mmbackupconfig 81
mmbuildgpl 83
mmcallhome 85
mmces 96
mmcesdr 105
mmchattr 113
mmchcluster 119
mmchconfig 123
mmchdisk 151
mmcheckquota 159
mmchfileset 163
mmchfs 169
mmchlicense 175
mmchmgr 178
mmchnode 180
mmchnodeclass 185
mmchnsd 188
mmchpolicy 191
mmchpool 194
mmchqos 196
mmclone 201
mmcrcluster 218
mmcrfileset 223
mmcrfs 229
mmcrnodeclass 239
mmcrnsd 241, 485, 763, 764
mmcrsnapshot 246
mmdefedquota 251
mmdefquotaoff 254
mmdefquotaon 257
mmdefragfs 260
mmdelacl 263
mmdelcallback 265
mmdeldisk 266, 764
mmdelfileset 271
mmdelfs 274
mmdelnode 276
mmdelnodeclass 279
mmdelnsd 281
mmdelsnapshot 283
mmdf 287
mmdiag 290
mmeditacl 295
mmedquota 298
mmexportfs 302
MMFS_FSSTRUCT 304
MMFS_SYSTEM_UNMOUNT 305
mmfsck 304
mmfsctl 313
mmgetacl 317
mmgetstate 320
mmhadoopctl 323
mmhealth 324, 340
mmimgbackup 329
mmimgrestore 333
mmimportfs 336
mmlinkfileset 350
mmlsattr 352
mmlscallback 355
mmlscluster 357
mmlsconfig 359
mmlsdisk 361
mmlsfileset 365
mmlsfs 369
mmlslicense 373
mmlsmgr 375
mmlsmount 377
mmlsnodeclass 379
mmlsnsd 381
mmlspolicy 384
mmlspool 386
mmlsqos 388
mmlsquota 391
mmlssnapshot 395, 661, 665
mmmigratefs 398
mmmount 400
mmnfs 402
mmnsddiscover 412
mmobj 414
mmperfmon query 428
mmpmon 438
mmprotocoltrace 443
mmpsnap 447
mmputacl 450
mmquotaoff 453
mmquotaon 455
mmremotecluster 457
mmremotefs 460
mmrepquota 463
mmrestoreconfig 467
mmrestorefs 471
mmrestripefile 475
mmrestripefs 478
mmrpldisk 485
mmsdrrestore 492
mmsetquota 494
mmshutdown 498
mmsmb 500
mmsnapdir 511, 638
mmstartup 515
mmtracectl 517
mmumount 521
mmunlinkfileset 524
mmuserauth 527
mmwinserv service
  managing 546
mmwinservctl 546
monitoring
  performance 438
mount point directory 230
mounting a file system 169
mtime 233, 711, 714, 717, 720, 736, 738, 740
multi-region object deployment
  mmobj command 414
multiple sessions 576

multiple-node environment 572, 601
model for DMAPI 601

N

Network Shared Disks (NSDs) 763
changing configuration attributes 188
creating 241
displaying 381
Network Shared Disks (NSDs), deleting 281
NFS (Network File System) 577
NFS V4 169, 233
NFS V4 ACL 170, 263, 295, 296, 317, 318, 450
nistCompliance attribute 138
node classes, user-defined
changing 185
creating 239
deleting 279
listing 379
node descriptor 29, 218
node designation 29, 218
node failure detection 134
node id 582
nodes
adding to a cluster 29
deleting from a cluster 276
noSpaceEventInterval attribute 139
NSD path 412
NSD server 336
NSD server list
changing 188
NSD server nodes
changing 188
choosing 241
NSD volume ID 241, 281
nsdBufSpace attribute 139
nsdCksumMismatch callback 19
nsdRAIDBufferPoolSizePct attribute 140
nsdRAIDTracks attribute 140
nsdServerWaitTimeForMount attribute 140
nsdServerWaitTimeWindowOnMount attribute 140
numaMemoryInterleave attribute 141

P

pagepool attribute 141
pagepoolMaxPhysMemPct attribute 141
parallel environment, DM applications 576
pdFailed callback 19
pdPathDown callback 19
pdRecovered callback 19
pdReplacePdisk callback 19
peer recovery cluster 313
Peer-to-Peer Remote Copy (PPRC) 313
performance 573
performance, monitoring 438
policy
applying 56
pool
displaying 386
postRGRelinquish callback 20
postRGTakeover callback 20
prefetchThreads attribute 141
preRGRelinquish callback 19
preRGTakeover callback 20
primary GPFS cluster configuration server 220
problem determination information, placement of 131
public/private key pair 67

Q

Quality of Service for I/O operations (QoS) level
changing 196
Quality of Service for I/O operations (QoS) settings
listing 388
quorum 314
quorum node 218, 276, 320
quota 577
quota files
replacing 159
quota information 730
quotas
activating 455
changing 298, 494, 727
checking 159
creating reports 463
deactivating 453
displaying 391
setting 298, 494

R

RAID stripe size 229
readReplicaPolicy attribute 142
rebalancing a file 475
rebalancing a file system 478
recovery
DODeferred deletions 604
mount event 604
synchronous event 603
unmount event 604
recovery groups
stanza files 338
refresh NSD server
mmnsddiscover 412
registering user event commands 10
release attribute 142
reliable DMAPI destroy events 574
remote copy command
changing 119
choosing 218
remote file systems 67
remote shell command
changing 119
choosing 218
replacing disks 485
replicated cluster 764
replication 170
querying 352
replication attributes
changing 113
replication factor 113
replication, strict 234
restoring configuration information 467
restoring NSD path
mmnsddiscover 412
restrictions
functions 583
restripeOnDiskFailure attribute 143
restriping a file 475
restriping a file system 478
rgOpenFailed callback 20

rgPanic callback 21
root credentials 583
rpcPerfNumberDayIntervals attribute 143
rpcPerfNumberHourIntervals attribute 143
rpcPerfNumberMinuteIntervals attribute 143
rpcPerfNumberSecondIntervals attribute 143
rpcPerfRawExecBufferSize attribute 143
rpcPerfRawStatBufferSize attribute 144

S

secondary GPFS cluster configuration server 220
semantic changes
  for the GPFS implementation 598
Server license 175
server node
  restoring NSD path 412
server node, NSD
  choosing 241
session
  failure 573, 599, 603
  recovery 603
session node 572, 598, 602
session, assuming a 572, 599
sessions
  description 572
  failure 572
  information string, changing 599
  maximum per node 572, 583
  state of 572
setup of interface for additional calls 698
shell script
  gpfsready 581
sidAutoMapRangeLength attribute 144
sidAutoMapRangeStart attribute 144
single-node 602
single-node environment 572, 601
SMB
  export 428
snapshot directory 653
snapshot handle 638, 642, 643, 645, 647, 649, 651, 655
  free 637
snapshot ID 638, 639, 643, 649
  comparing 620
  internal 661, 665
snapshot name 645, 655
snapshots
  coexistence 571
  creating 246
  deleting 246, 283
  directory 246
  displaying 395
  fileset 246
  global 246
  listing 395
  restoring a file system 471
  restoring a fileset 471
sort-command parameter of mmapplypolicy command 63
source node 572, 602
sparse file 694
spectrumscale 548
spectrumscale installation toolkit 548
stanza files
  recovery group 338
starting GPFS 515
storage
  pre-allocating 723
storage pool 386
storage pool properties
  changing 194
storage pools
  ID 675
  name 675
strict replication 170, 234
structure
  dm_eventmsg 582
  dm_mount_event 575, 582, 599
  dm_namesp_event 582
  dm_region_t 582
  dm_stat 582
  dm_stat_t 599
  dm_vardata_t 582
  uio 601
structures
  gpfs_acl_t 610
  gpfs_direntx_t 624
  gpfs_direntx64_t 626
  gpfs_fssnap_handle_t 638
  gpfs_fssnap_id_t 639
  gpfs_iattr_t 659
  gpfs_iattr64_t 662
  gpfs_ifile_t 668, 669, 671
  gpfs_iscan_t 697
  gpfs_opaque_acl_t 710
  gpfs_quotaInfo_t 730
  gpfsFcntlHeader_t 746
  gpfsGetFilesetName_t 747
  gpfsGetReplication_t 748
  gpfsGetSetXAttr_t 750
  gpfsGetSnapshotName_t 752
  gpfsGetStoragePool_t 753
  gpfsListXAttr_t 754
  gpfsRestripeData_t 755
  gpfsSetReplication_t 757
  gpfsSetStoragePool_t 759
subnets attribute 144
subroutine
  gpfs_close_inodescan() 619
  gpfs_cmp_fssnapid() 620
subroutines
  gpfs_clone_copy() 611
  gpfs_clone_snap() 613
  gpfs_clone_split() 615
  gpfs_clone_unsnap() 617
  gpfs_declone() 622
  gpfs_fcntl() 628
  gpfs_fgetattrs() 631
  gpfs_fputattrs() 633
  gpfs_fputattrswithpathname() 635
  gpfs_free_fssnaphandle() 637
  gpfs_fstat() 640
  gpfs_get_fsname_from_fssnaphandle() 642
  gpfs_get_fssnaphandle_by_fssnapid() 643
  gpfs_get_fssnaphandle_by_name() 645
  gpfs_get_fssnaphandle_by_path() 647
  gpfs_get_fssnapid_from_fssnaphandle() 649
  gpfs_get_pathname_from_fssnaphandle() 651
  gpfs_get_snapdirname() 653
  gpfs_get_snapname_from_fssnaphandle() 655
  gpfs_getacl() 657
  gpfs_iclose() 666
  gpfs_igetattrs() 669
  gpfs_igetattrsx() 671
  gpfs_igetfilesetname() 673

subroutines *(continued)*
gpfs_igetstoragepool() 675
gpfs_iopen() 677
gpfs_iopen64() 679
gpfs_iputattrsx() 681
gpfs_iread() 684
gpfs_ireaddir() 686
gpfs_ireaddir64() 688
gpfs_ireadlink() 690
gpfs_ireadlink64() 692
gpfs_ireadx() 694
gpfs_lib_init() 698
gpfs_lib_term() 699
gpfs_next_inode_with_xattrs() 704
gpfs_next_inode_with_xattrs64() 706
gpfs_next_inode() 700
gpfs_next_inode64() 702
gpfs_next_xattr() 708
gpfs_open_inodescan_with_xattrs() 717
gpfs_open_inodescan_with_xattrs64() 720
gpfs_open_inodescan() 711
gpfs_open_inodescan64() 714
gpfs_prealloc() 723
gpfs_putacl() 725
gpfs_quotactl() 727
gpfs_seek_inode() 732
gpfs_seek_inode64() 734
gpfs_stat_inode_with_xattrs() 742
gpfs_stat_inode_with_xattrs64() 744
gpfs_stat_inode() 738
gpfs_stat_inode64() 740
gpfs_stat() 736
symbolic link
reading 690, 692
syncFSconfig 313
system snapshots 5
systemLogLevel attribute 145

T

tiebreakerDisks attribute 145
timeout period 498
token, usage 575
tokens
input arguments 600
trace-recycle
changing 518
tracing attributes, changing 517
traditional ACL 170, 295, 296, 317, 318
traditional ACLs
NFS V4 ACL 233
Windows 233
transparent cloud tiering
commands
mmcloudgateway 204

U

UID domain 220
uidDomain attribute 146
unified file and object access
mmobj command 414
unlinking
filesets 524
unmountOnDiskFail attribute 146
usage restrictions 583
usePersistentReserve attribute 146
user event commands, registering 10
user exit
GPFS 764
IBM Spectrum Scale 764
user exits 761
GPFS 763
IBM Spectrum Scale 763
mmsdrbackup 762
nsddevices 763
syncfsconfig 764
user quota 252, 254, 257, 299, 391, 455, 463
user space buffer 113
user-defined callbacks
deleting 265
listing 355
user-defined node classes
changing 185
creating 239
deleting 279
listing 379
using the gpfs.snap command
gathering data 5

V

verbsPorts attribute 147
verbsRdma attribute 147
verbsRdmaCm attribute 147
verbsRdmaRoCEToS attribute 147
verbsRdmaSend attribute 148
verbsRdmasPerConnection attribute 148
verbsRdmasPerNode attribute 148
verbsSendBufferMemoryMB attribute 148

W

worker1Threads attribute 148

X

XDSM standard 570, 572, 601, 602

Product Number: 5725-Q01
5641-GPF
5725-S28

Printed in USA

SA23-1456-01