IBM Spectrum Scale
Version 4 Release 2.1

# *Administration Guide*

SA23-1455-01

IBM Spectrum Scale
Version 4 Release 2.1

# *Administration Guide*

SA23-1455-01

**Note**
Before using this information and the product it supports, read the information in "Notices" on page 629.

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 xiv. When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

# Contents

# Tables

# 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:<br>• Use the commands, programming interfaces, and user exits unique to GPFS<br>• Manage clusters, file systems, disks, and quotas<br>• Export a GPFS file system using the Network File System (NFS) protocol | System administrators or programmers of GPFS systems |

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

| Information unit | Type of information | Intended users |
|---|---|---|
| *IBM Spectrum Scale: Administration Guide* | This information unit explains how to use the following advanced features of GPFS:<br>• Accessing GPFS file systems from other GPFS clusters<br>• Policy-based data management for GPFS<br>• Creating and maintaining snapshots of GPFS file systems<br>• Establishing disaster recovery for your GPFS cluster<br>• Monitoring GPFS I/O performance with the **mmpmon** command<br>• 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:<br>• Introducing GPFS<br>• GPFS architecture<br>• Planning concepts for GPFS<br>• Installing GPFS<br>• Migration, coexistence and compatibility<br>• Applying maintenance<br>• Configuration and tuning<br>• 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.<br><br>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.<br><br>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.<br><br>Use this information if you intend to write application programs to do the following:<br>• Monitor events associated with a GPFS file system or with an individual file<br>• 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 xiv describes the typographic conventions used in this information. UNIX file name conventions are used throughout this information.

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

*Table 2. Conventions*

| Convention | Usage |
|---|---|
| **bold** | **Bold** words or characters represent system elements that you must use literally, such as commands, flags, values, and selected menu options.<br><br>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.<br><br>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.<br><br>*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:<br>`mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \`<br>`-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*.<br><br>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*.

**/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<br><br>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<br><br>(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<br>2. IBM Spectrum Scale: Administration and Programming Reference<br>3. IBM Spectrum Scale: Advanced Administration Guide<br>4. IBM Spectrum Scale: Data Management API Guide<br>5. IBM Spectrum Scale: Problem Determination Guide | 1. Summary of changes<br>2. Quick reference<br>3. Product overview<br>4. Planning<br>5. Installing and upgrading<br>6. Configuring<br>7. Administering<br>8. Monitoring<br>9. Troubleshooting<br>10. Command reference<br>11. Programming reference<br>12. Library and related publications<br>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. Configuring the GPFS cluster

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

GPFS cluster management tasks include the following.

- "Creating your GPFS cluster"
- "Displaying GPFS cluster configuration information"
- "Specifying nodes as input to GPFS commands" on page 67
- "Adding nodes to a GPFS cluster" on page 2
- "Deleting nodes from a GPFS cluster" on page 3
- "Changing the GPFS cluster configuration data" on page 4
- "Node quorum considerations" on page 19
- "Node quorum with tiebreaker considerations" on page 19
- "Displaying and changing the file system manager node" on page 20
- "Determining how long **mmrestripefs** takes to complete" on page 20
- "Starting and stopping GPFS" on page 21

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

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

## Creating your GPFS cluster

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

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

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

## Displaying GPFS cluster configuration information

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

The command displays:

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

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

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

To display this information, enter:
```
mmlscluster
```

The system displays information similar to:
```
GPFS cluster information
========================
  GPFS cluster name:         cluster1.kgn.ibm.com
  GPFS cluster id:           680681562214606028
  GPFS UID domain:           cluster1.kgn.ibm.com
  Remote shell command:      /usr/bin/ssh
  Remote file copy command:  /usr/bin/scp
  Repository type:           CCR

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

For complete usage information, see **mmlscluster command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Adding nodes to a GPFS cluster

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

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

To add node **k164n06** to the GPFS cluster, enter:
```
mmaddnode -N k164n06
```

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

To confirm the addition of the nodes, enter:

```
mmlscluster
```

The system displays information similar to:

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

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

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

| You can also use the **spectrumscale** installation toolkit to add nodes. For more information, see *Adding
| nodes, NSDs, or file systems to an installation process* in *IBM Spectrum Scale: Concepts, Planning, and
| Installation Guide*.

For complete usage information, see **mmaddnode command**, **mmlscluster command** and **mmchlicense command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting nodes from a GPFS cluster

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

You must follow these rules when deleting nodes:

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

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

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

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

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

To confirm the deletion of the nodes, issue:

```
mmlscluster
```

The system displays information similar to:

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

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

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

| For information on deleting protocol nodes (CES nodes) from a cluster, see "Deleting a Cluster Export
| Services node from an IBM Spectrum Scale cluster" on page 27.

For complete usage information, see **mmdelnode command** and **mmlscluster command** in *IBM Spectrum Scale: Command and Programming Reference*.

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

## Changing the GPFS cluster configuration data

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

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

- *mmchcluster command* in *IBM Spectrum Scale: Command and Programming Reference*
- *mmchconfig command* in *IBM Spectrum Scale: Command and Programming Reference*

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

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

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

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

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

- Synchronize the primary GPFS cluster configuration server node. If an invocation of the **mmchcluster** command fails, you will be prompted to reissue the command and specify **LATEST** on the **-p** option to synchronize all of the nodes in the GPFS cluster. Synchronization instructs all nodes in the GPFS cluster to use the most recently specified primary GPFS cluster configuration server.

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

```
mmchcluster -p k164n06
```

The system displays information similar to:

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

To confirm the change, enter:

```
mmlscluster
```

The system displays information similar to:

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

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

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

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

For complete usage information, see **mmchcluster command** and **mmlscluster command** in *IBM Spectrum Scale: Command and Programming Reference*

You might be able to tune your cluster for better performance by reconfiguring one or more attribute. Before you change any attribute, consider how the changes will affect the operation of GPFS. For a detailed discussion, see *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and *mmcrcluster command* in *IBM Spectrum Scale: Command and Programming Reference* guide.

Table 5 on page 6 details the GPFS cluster configuration attributes which can be changed by issuing the **mmchconfig** command. Variations under which these changes take effect are noted:

1. Take effect immediately and are permanent (**-i**).

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

*Table 5. Configuration attributes on the* **mmchconfig** *command*

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

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **cnfsVersions**<br>List of protocol versions that CNFS should start and monitor | yes | no | yes | no | immediately |
| **dataDiskCacheProtectionMethod**<br>Defines the cache protection method for disks that are used for the GPFS file system. | no | no | yes | no | on restart of the daemon |
| **dataDiskWaitTimeForRecovery**<br>Controls the suspension of **dataOnly** disk recovery | yes | no | no | yes | immediately |
| **dataStructureDump**<br>Path for the storage of dumps | yes | no | no | yes | if not immediately, on restart of the daemon |
| **deadlockBreakupDelay**<br>When to attempt breaking up a detected deadlock | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockDataCollectionDailyLimit**<br>Maximum number of times to collect debug data in 24 hours | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockDataCollectionMinInterval**<br>Minimum interval between two consecutive collections of debug data | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockDetectionThreshold**<br>Threshold for detecting deadlocks | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockDetectionThresholdForShortWaiters**<br>Threshold for detecting deadlocks from short waiters | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockDetectionThresholdIfOverloaded**<br>Threshold for detecting deadlocks when a cluster is overloaded | yes | yes | no | no | immediately with **-i** or **-I** |
| **deadlockOverloadThreshold**<br>Threshold for detecting cluster overload | yes | yes | no | no | immediately with **-i** or **-I** |
| **debugDataControl**<br>Controls the amount of debug data collected | yes | no | no | yes | immediately |
| **defaultMountDir**<br>Default parent directory for GPFS file systems | yes | yes | no | no | for new file systems |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **disableInodeUpdateOnFdatasync**<br><br>Controls inode update on **fdatasync** for **mtime** and **atime** updates. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **dmapiDataEventRetry**<br><br>DMAPI attribute | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **dmapiEventTimeout**<br><br>DMAPI attribute | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **dmapiMountEvent**<br><br>DMAPI attribute | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **dmapiMountTimeout**<br><br>DMAPI attribute | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **dmapiSessionFailureTimeout**<br><br>DMAPI attribute | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **enableIPv6**<br><br>Controls whether the GPFS daemon is to communicate through the IPv6 network. | no | no | only when **enableIPv6** is set to **yes** | not applicable | if not immediately, on restart of the daemon |
| **enforceFilesetQuotaOnRoot**<br><br>Controls fileset quota settings for the root user | yes | yes | no | no | if not immediately, on restart of the daemon |
| **expelDataCollectionDailyLimit**<br><br>Maximum number of times to collect expel-related debug data in 24 hours | yes | yes | no | no | immediately with **-i** or **-I** |
| **expelDataCollectionMinInterval**<br><br>Minimum interval between two consecutive collections of expel-related debug data | yes | yes | no | no | immediately with **-i** or **-I** |
| **failureDetectionTime**<br><br>Indicates the amount of time it will take to detect that a node has failed | no | no | yes | no | on restart of the daemon |
| **fastestPolicyCmpThreshold**<br><br>Indicates the disk comparison count threshold, above which GPFS forces selection of this disk as the preferred disk to read | yes | yes | no | yes | immediately with -i |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **fastestPolicyMaxValidPeriod**<br><br>Indicates the time period after which the disk's current evaluation is considered invalid | yes | yes | no | yes | immediately with -i |
| **fastestPolicyMinDiffPercent**<br><br>A percentage value indicating how GPFS selects the fastest between two disks | yes | yes | no | yes | immediately with -i |
| **fastestPolicyNumReadSamples**<br><br>Controls how many read samples taken to evaluate the disk's recent speed | yes | yes | no | yes | immediately with -i |
| **fileHeatLossPercent**<br><br>Specifies the reduction rate of **FILE_HEAT** value for every **fileHeatPeriodMinutes** of file inactivity. | yes | yes | no | no | if not immediately, on restart of the daemon |
| **fileHeatPeriodMinutes**<br><br>Specifies the inactivity time before a file starts to lose **FILE_HEAT** value. | yes | yes | no | no | if not immediately, on restart of the daemon |
| **FIPS1402mode**<br><br>Controls whether GPFS operates in FIPS 140-2 mode. | no | no | no | not applicable | on restart of the daemon |
| **forceLogWriteOnFdatasync**<br><br>Controls forcing log writes to disk. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **ignorePrefetchLUNCount**<br><br>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**. | yes | yes | no | yes | immediately with **-i** |
| **lrocData**<br><br>Controls whether user data will be populated into the local read-only cache. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **lrocDataMaxFileSize**<br><br>Limits the data that may be saved in the local read-only cache to only the data from small files. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **lrocDataStubFileSize**<br><br>Limits the data that may be saved in the local read-only cache to only the data from the first portion of all files. | yes | yes | no | yes | immediately with **-i** or **-I** |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **lrocDirectories**<br>Controls whether directory blocks will be populated into the local read-only cache. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **lrocInodes**<br>Controls whether inodes from open files will be populated into the local read-only cache. | yes | yes | no | yes | immediately with **-i** or **-I** |
| **maxblocksize**<br>Maximum file system block size allowed | no | no | no | yes | on restart of the daemon |
| **maxBufferDescs**<br>Can be tuned to cache very large files | no | no | no | yes | on restart of the daemon |
| **maxDownDisksForRecovery**<br>Maximum number of failed disks allowed for automatic recovery to continue | yes | no | no | yes | immediately |
| **maxFailedNodesForRecovery**<br>Maximum number of unavailable nodes allowed before automatic disk recovery is cancelled | yes | no | no | yes | immediately |
| **maxFcntlRangesPerFile**<br>Specifies the number of **fcntl** locks that are allowed per file | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **maxFilesToCache**<br>Number of inodes to cache for recently used files | no | no | no | yes | on restart of the daemon |
| **maxMissedPingTimeout**<br>Handles high network latency in a short period of time | no | no | no | no | on restart of the daemon |
| **maxMBpS**<br>I/O throughput estimate | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **maxStatCache**<br>Number of inodes to keep in stat cache | no | no | no | yes | on restart of the daemon |
| **metadataDiskWaitTimeForRecovery**<br>Controls the suspension of metadata disk recovery | yes | no | no | yes | immediately |
| **minDiskWaitTimeForRecovery**<br>Controls the suspension of disk recovery | yes | no | no | yes | immediately |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **minMissedPingTimeout**<br><br>Handles high network latency in a short period of time | no | no | no | no | on restart of the daemon |
| **mmapRangeLock**<br><br>Specifies POSIX or non-POSIX **mmap** byte-range semantics<br><br>**Note:** The list of *NodeName*s is allowed, but it is not recommended. | yes | yes | no | yes | immediately |
| **nfsPrefetchStrategy**<br><br>Optimizes prefetching for NFS file-style access patterns | yes | yes | no | yes | immediately with **-i** |
| **nistCompliance**<br><br>Controls whether GPFS operates in NIST 800-131A mode for security transport mechanisms. | no | no | no | not applicable | if not immediately, on restart of the daemon |
| **noSpaceEventInterval**<br><br>Time interval between **noDiskSpace** events of a file system | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **nsdBufSpace**<br><br>Percentage of the pagepool reserved for the network transfer of NSD requests | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **nsdInlineWriteMax**<br><br>Specifies the maximum transaction size that can be sent as embedded data in an NSD-write RPC | yes | yes | no | yes | immediately with **-i** |
| **nsdMaxWorkerThreads**<br><br>Sets the maximum number of NSD threads on an NSD server that concurrently transfers data with NSD clients | no | no | no | yes | on restart of the daemon |
| **nsdMinWorkerThreads**<br><br>Used to increase the NSD server performance by providing a large number of dedicated threads for NSD service | no | no | no | yes | on restart of the daemon |
| **nsdMultiQueue**<br><br>Sets the number of queues | yes | yes | no | yes | immediately with **-i** |
| **nsdRAIDBufferPoolSizePct**<br><br>Percentage of the page pool that is used for the IBM Spectrum Scale RAID vdisk buffer pool | yes | yes | no | yes | if not immediately, on restart of the daemon |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **nsdRAIDTracks**<br><br>Number of tracks in the IBM Spectrum Scale RAID buffer pool | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **nsdServerWaitTimeForMount**<br><br>Number of seconds to wait for an NSD server to come up | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **nsdServerWaitTimeWindowOnMount**<br><br>Time window to determine if quorum is to be considered *recently formed* | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **numaMemoryInterleave** | no | no | no | yes | on restart of the daemon |
| **pagepool**<br><br>Size of buffer cache on each node | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **pagepoolMaxPhysMemPct**<br><br>Percentage of physical memory that can be assigned to the page pool | no | no | no | yes | on restart of the daemon |
| **pitWorkerThreadsPerNode**<br><br>Maximum number of threads to be involved in parallel processing on each node serving as a Parallel Inode Traversal (PIT) worker | yes | yes | no | yes | immediately with **-i** or **-I** |
| **prefetchPct**<br><br>Acts as a guideline to limit the pagepool space that is to be used for prefetch and write-behind buffers for active sequential streams | no | no | no | yes | on restart of the daemon |
| **prefetchThreads**<br><br>Maximum number of threads dedicated to prefetching data | no | no | no | yes | on restart of the daemon |
| **readReplicaPolicy**<br><br>The disk read replica policy | yes | yes | no | yes | immediately with -i |
| **release=LATEST**<br><br>Complete the migration to a new release | yes | no | no | no | if not immediately, on restart of the daemon |
| **restripeOnDiskFailure**<br><br>Specifies whether GPFS will attempt to automatically recover from certain common disk failure situations. | yes | no | no | yes | immediately with **-i** |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **rpcPerfNumberDayIntervals**<br><br>Number of days that aggregated RPC data is saved | no | no | no | yes | on restart of the daemon |
| **rpcPerfNumberHourIntervals**<br><br>Number of hours that aggregated RPC data is saved | no | no | no | yes | on restart of the daemon |
| **rpcPerfNumberMinuteIntervals**<br><br>Number of minutes that aggregated RPC data is saved | no | no | no | yes | on restart of the daemon |
| **rpcPerfNumberSecondIntervals**<br><br>Number of seconds that aggregated RPC data is saved | no | no | no | yes | on restart of the daemon |
| **rpcPerfRawExecBufferSize**<br><br>The buffer size of the raw RPC execution times | no | no | no | yes | on restart of the daemon |
| **rpcPerfRawStatBufferSize**<br><br>The buffer size of the raw RPC statistics | no | no | no | yes | on restart of the daemon |
| **seqDiscardThreshold**<br><br>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. | yes | yes | no | yes | immediately with **-i** |
| **sidAutoMapRangeLength**<br><br>Controls the length of the reserved range for Windows SID to UNIX ID mapping | yes | yes | no | no | if not immediately, on restart of the daemon |
| **sidAutoMapRangeStart**<br><br>Specifies the start of the reserved range for Windows SID to UNIX ID mapping | no | no | no | no | on restart of the daemon |
| **syncbuffsperiteration**<br><br>Used to expedite buffer flush and the rename operations done by MapReduce jobs. | yes | yes | no | yes | immediately with **-i** |
| **systemLogLevel**<br><br>Filters messages sent to the system log on Linux | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **subnets**<br><br>List of subnets to be used for most efficient daemon-to-daemon communication | no | no | no | yes | on restart of the daemon |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **tiebreakerDisks** (CCR repository)<br><br>List of tiebreaker disks (NSDs) | no | no | no | no | immediately |
| **tiebreakerDisks** (server-based repository)<br><br>List of tiebreaker disks (NSDs) | no | no | yes | no | on restart of the daemon |
| **uidDomain**<br><br>The UID domain name for the cluster. | no | no | yes | no | on restart of the daemon |
| **unmountOnDiskFail**<br><br>Unmount the file system on a disk failure | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **usePersistentReserve**<br><br>Enables or disables persistent reserve (PR) on the disks | no | no | yes | no | on restart of the daemon |
| **verbsPorts**<br><br>Specifies InfiniBand device names and port numbers | no | no | no | yes | on restart of the daemon |
| **verbsRdma**<br><br>Enables or disables InfiniBand RDMA using the Verbs API. | no | no | no | yes | on restart of the daemon |
| **verbsRdmaCm**<br><br>Enables or disables InfiniBand RDMA_CM using the RDMA_CM API. | no | no | no | yes | on restart of the daemon |
| **verbsRdmaRoCEToS**<br><br>Specifies the Type of Service (ToS) value for clusters using RDMA over Converged Ethernet (RoCE). | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **verbsRdmaSend**<br><br>Enables or disables the use of InfiniBand RDMA rather than TCP for most GPFS daemon-to-daemon communication. | no | no | no | yes | on restart of the daemon |
| **verbsRdmasPerConnection**<br><br>Sets the maximum number of RDMAs allowed per connection. | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **verbsRdmasPerNode**<br><br>Sets the maximum number of RDMAs allowed per node. | yes | yes | no | yes | if not immediately, on restart of the daemon |

*Table 5. Configuration attributes on the* **mmchconfig** *command (continued)*

| Attribute name and Description | -i option allowed | -I option allowed | GPFS must be stopped on all nodes | List of *NodeName*s allowed | Change takes effect |
|---|---|---|---|---|---|
| **verbsSendBufferMemoryMB**<br><br>Sets the amount of page pool memory to reserve as dedicated buffer space for use by **verbsRdmaSend**. | yes | yes | no | yes | if not immediately, on restart of the daemon |
| **workerThreads**<br><br>Sets an integrated group of variables that tune file system performance. | no | no | no | yes | on restart of the daemon |
| **worker1Threads**<br><br>Sets the maximum number of concurrent file operations | yes<br><br>(only when adjusting value down) | yes<br><br>(only when adjusting value down) | no | yes | on restart of the daemon |
| **writebehindThreshold**<br><br>Specifies the point at which GPFS starts flushing new data out of the pagepool for a file that is being written sequentially. | yes | yes | no | yes | immediately with **-i** |

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

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

The system displays information similar to:

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

For complete usage information, see **mmchconfig command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Security mode

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

There are three security modes:

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

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

*Cipher*
: To set this mode, you must specify the name of a supported cipher, such as `AES128-GCM-SHA256`.

The sending and receiving nodes authenticate each other with a TLS handshake. A TLS connection is established. The transmitted data is encrypted with the specified cipher and is checked for data integrity.

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

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

  ```
  mmauth show ciphers
  ```

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

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

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

```
mmlsconfig cipherlist
```

You can change the security mode with the following command:

```
mmchconfig cipherlist=security_mode
```

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

The default security mode is **EMPTY** in IBM Spectrum Scale V4.1 or earlier and is **AUTHONLY** in IBM Spectrum Scale V4.2 or later. If you migrate a cluster from IBM Spectrum Scale V4.1 to V4.2 or later by running `mmchconfig release=LATEST`, the command checks the security mode. If the mode is **EMPTY**, the command fails with an error message. You then can do either of two actions:

- Change the security mode to a valid value other than **EMPTY**, such as **AUTHONLY**, and rerun the `mmchconfig release=LATEST` command. Or,
- Leave the security mode set to **EMPTY** and re-run the `mmchconfig release=LATEST` command with the option `--accept-empty-cipherlist-security`.

For more information, see *Completing the migration to a new level of IBM Spectrum Scale* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

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

## Running IBM Spectrum Scale without remote root login

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

Every administration node in the IBM Spectrum Scale cluster must be able to run the administration commands, generally known as *mm commands*, on any other node in the cluster. Each administration node must be able to do so without the use of a password and without producing any extraneous messages. Also, most of the IBM Spectrum Scale administration commands must run at the root level. One solution

to meet these requirements is to configure each node to permit general remote login to its root user ID. However, there are secure solutions available that do not require root-level login.

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

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

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

## Configuring sudo

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

Perform the following steps before you start configuring sudo:

1. Create a user and group to run administration commands.

   **Note:** The examples in this section have the user name `gpfsadmin` and the group `gpfs`.
2. Allow the root user from an administration node to execute commands on all nodes including the current node as the `gpfsadmin` user id without being prompted for a password.
3. Install the sudo program. Sudo is a free open source software that is distributed under a license.

Do the following steps on each node in the cluster:

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

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

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

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

   The first line preserves the environment variables that the IBM Spectrum Scale administration commands need to run. The second line allows the users in the `gpfs` group to run administration commands without being prompted for a password. The third line disables `requiretty`. When this flag is enabled, sudo blocks the commands that do not originate from a TTY session.
3. Perform the following steps to verify that the `sshwrap` and `scpwrap` scripts work correctly.
   a. `sshwrap` is an IBM Spectrum Scale sudo wrapper script for the remote shell command that is installed with IBM Spectrum Scale. To verify that it works correctly, run the following command as the *gpfsadmin* user:

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

      **Note:** `nodeName` is the name of an IBM Spectrum Scale node in the cluster
   b. `scpwrap` is an IBM Spectrum Scale sudo wrapper script for the remote file copy command that is installed with IBM Spectrum Scale. To verify that it works correctly, run the following command as the *gpfsadmin* user:

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

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

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

## Configuring the cluster to use sudo wrapper scripts

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

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

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

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

     ```
     gpfsadmin@c13c1apv7 admin]$mmlscluster
     GPFS cluster information
     ========================
     GPFS cluster name: c13c1apv7.gpfs.net
     GPFS cluster id: 12275146245716580740
     GPFS UID domain: c13c1apv7.gpfs.net
     Remote shell command: sudo wrapper in use
     Remote file copy command: sudo wrapper in use
     Repository type: CCR
     Node Daemon node name IP address Admin node name Designation
     -----------------------------------------------------------
     1 c13c1apv7.gpfs.net 192.168.148.117 c13c1apv7.gpfs.net quorum
     2 c13c1apv8.gpfs.net 192.168.148.118 c13c1apv8.gpfs.net
     ```
- To configure an existing cluster to call the sudo wrapper scripts, use these steps.
  1. Log in with the user ID. This example uses gpfsadmin as the user ID.
  2. Issue the **mmchcluster** command with the**--use-sudo-wrapper** option to start using the sudo wrappers:

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

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

## Configuring IBM Spectrum Scale GUI to use sudo wrapper

To use the IBM Spectrum Scale management GUI on a cluster where sudo wrappers are used, you need to make the following configuration changes:

1. Change *GPFS_ADMIN=root* to the user name that you configured as the sudo user in the `gpfsgui.properties` file that is located at: `/usr/lpp/mmfs/gui/conf`. For example, *GPFS_ADMIN=gpfsadmin*.
2. Issue the **systemctl restart gpfsgui** command to restart the GUI.

**Note:** If sudo wrappers are enabled on the cluster but GUI is not configured for it, the system raises an event.

## Configuring a cluster to stop using sudo wrapper scripts

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

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

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

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

## Node quorum considerations

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

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

## Node quorum with tiebreaker considerations

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

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

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

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

Consider these points:

- You are not permitted to change a GPFS cluster configuration to use node quorum with tiebreaker if there are more than eight existing quorum nodes.
- You can have a maximum of three tiebreaker disks.
- The disks must be directly attached to all quorum nodes.
- When adding tiebreaker disks:
  - If the tiebreaker disks are part of a file system, GPFS should be up and running.
  - If the tiebreaker disks are not part of a file system, GPFS can be either running or shut down.
- When using the traditional server-based (non-CCR) configuration repository, the GPFS daemons must be down on all nodes in the cluster when running **mmchconfig tiebreakerDisks**.

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

```
mmchconfig tiebreakerDisks=DEFAULT
```

## Displaying and changing the file system manager node

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

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

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

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

```
mmlsmgr fs1
```

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

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

For complete usage information, see **mmlsmgr command** in *IBM Spectrum Scale: Command and Programming Reference*.

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

```
mmchmgr fs1 k145n32
```

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

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

For complete usage information, see **mmchmgr command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Determining how long `mmrestripefs` takes to complete

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

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

1. How much data potentially needs to be moved. You can estimate this using the **df** command.
2. How many GPFS client nodes there are to do the work.
3. How much Network Shared Disk (NSD) server bandwidth is available for I/O.

4. If you have added new disks to a file system, use the **mmdf** command to determine how much free space is available on each of the new disks.

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

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

```
2 * fileSystemSize / averageDiskserverDataRate
```

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

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

## Starting and stopping GPFS

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

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

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

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

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

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

```
mmstartup  -a
```

The output is similar to this:

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

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

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

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

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

For complete usage information, see **mmstartup command** in *IBM Spectrum Scale: Command and Programming Reference*.

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

```
mmshutdown -a
```

The system displays information similar to:

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

For complete usage information, see **mmshutdown command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Shutting down an IBM Spectrum Scale cluster

Use the following information to shut down an IBM Spectrum Scale cluster in an emergency situation.

1. Stop the protocol services on all protocol nodes in the cluster using the **mmces service stop** command. For example:

   ```
   mmces service stop nfs -a
   mmces service stop smb -a
   mmces service stop obj -a
   ```

2. Unmount all file systems, except the CES shared root file system, on all nodes in the cluster using the **mmumount** command.
3. Stop GPFS daemons on all protocol nodes in the cluster using the **mmshutdown -N cesNodes** command.
4. Unmount all file systems on all nodes in the cluster using the **mmumount all -a** command.
5. Stop GPFS daemons on all nodes in the cluster using the **mmshutdown -a** command.

After performing these steps, depending on your operating system, shut down your servers accordingly.

Before shutting down and powering up your servers, consider the following:

- You must shut down NSD servers before the storage subsystem. While powering up, the storage subsystem must be online before NSD servers are up so that LUNs are visible to them.
- In a power-on scenario, verify that all network and storage subsystems are fully operational before bringing up any IBM Spectrum Scale nodes.
- On the Power® platform, you must shut down operating systems for LPARs first and then power off servers using Hardware Management Console (HMC). HMC must be the last to be shut down and the first to be powered up.
- It's preferable to shut down your Ethernet and InfiniBand switches using the management console instead of powering them off. In any case, network infrastructure such as switches or extenders must be powered off last.
- After starting up again, verify that functions such as AFM, policies, etc. are operational. You might need to manually restart some functions.
- There are a number other GPFS functions that could be interrupted by a shutdown. Ensure that you understand what else might need to be verified depending on your environment.

# Chapter 2. Configuring the CES and protocol configuration

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

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

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

## Configuring Cluster Export Services

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

For more information on the CES features, see Chapter 28, "Implementing Cluster Export Services," on page 393.

## Setting up Cluster Export Services shared root file system

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

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

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

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

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

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

To list the current cesSharedRoot, run

```
mmlsconfig cesSharedRoot
cesSharedRoot /gpfs/gpfs-ces/
```

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

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

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

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

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

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

## Configuring Cluster Export Services nodes

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

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

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

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

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

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

```
mmces node list
```

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

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

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

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

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

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

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

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

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

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

## Configuring CES protocol service IP addresses

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

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

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

After adding all desired CES protocol service IP addresses, verify the configuration:

```
mmces address list
```

Use **mmces address add --ces-ip 192.168.6.6** to add an IP address to the CES IP address pool. The IP address will be assigned to a CES node according to the CES "Address distribution policy".

CES addresses may be assigned to CES groups. A CES group is identified by a group name consisting of alphanumeric characters which are case-sensitive. Addresses may be assigned to a group when they are defined by issuing the following command:

```
mmces address add --ces-ip 192.168.6.6 --ces-group group1
```

The group assignment may be changed by issuing the following command:

```
mmces address change --ces-ip 192.168.6.6 --ces-group group2
```

The group assignment may be removed by issuing the following command:

```
mmces address change --ces-ip 192.168.6.6 --remove-group
```

A CES address which is associated with a group may only be assigned to a node which is also associated with the same group. A node may belong to multiple groups while an address may not.

As an example, consider a configuration with three nodes - all of which are able to host addresses on subnet A and two of the nodes are able to host addresses on subnet B. The nodes must have existing non-CES IP address of the same subnet configured on the interfaces intended to be used for the CES IPs. Also four addresses are defined, two on each subnet.

Node1: groups=subnetA,subnetB

Node2: groups=subnetA,subnetB

Node3: groups=subnetA

Address1: subnetA

Address2: subnetA

Address3: subnetB

Address4: subnetB

In this example, Address1 and Address2 may be assigned to any of the three nodes, but Address3 and Address4 may be assigned to only Node1 or Node2.

If an address is assigned to a group for which there are no healthy nodes, the address will remain unassigned until a node in the same group becomes available.

Addresses without a group assignment may be assigned to any node.

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

## Deploying Cluster Export Services packages on existing IBM Spectrum Scale 4.1.1 and later nodes

Use the following instructions to copy packages on your protocol nodes and to deploy these packages.

1. Copy the required packages to the protocol node from the location where the self-extracting package was extracted.

   By default, installation images are extracted to the target directory `/usr/lpp/mmfs/4.2.1.0`
2. Install packages by issuing the following command: **rpm -ivh** *Package_Name1 Package_Name2* **...** *Package_NameN*

For a list of packages applicable for the current IBM Spectrum Scale release, see *Manually installing the software packages on Linux nodes* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

3. Set the server licenses for each CES node by issuing the following command: **mmchlicense server --accept -N ces_node_ips**

   For example:

   ```
   mmchlicense server --accept -N 203.0.113.7,203.0.113.9
   ```

4. Enable CES by issuing the following command: **mmchnode -N ces_nodes --ces-enable**

   Foe example:

   ```
   mmchnode -N 203.0.113.7,203.0.113.9 --ces-enable
   ```

5. Assign export IPs for each export_IP by issuing this command: **mmces address add --ces-ip export_IP**

## Verifying the final CES configurations

After you finish the Cluster Export Services (CES) configuration steps, verify the final configuration.

To verify your configuration, run the following command:

```
mmlscluster --ces
```

For more information, see the **mmces node list** and **mmces address list** options in *mmces command* in *IBM Spectrum Scale: Command and Programming Reference*.

# Creating and configuring file systems and filesets for exports

If you have not done so previously, create the file systems and the filesets for the data to be exported through the protocol services. For more information, see *mmcrfs* and *mmcrfileset* in *IBM Spectrum Scale: Command and Programming Reference*.

### Creating a fileset through the GPFS GUI

To create a fileset, log on to the IBM Spectrum Scale GUI and select **Files** > **Filesets** > **Create Fileset**.

IBM strongly recommends to configure the file systems to only allow NFSv4 ACLs through the -k nfs4 option for mmcrfs. When using the default configuration profiles (/usr/lpp/mmfs/profiles) that are included with IBM Spectrum Scale, the NFSv4 ACL setting is already set from the profile configuration (see "Authorizing file protocol users" on page 258 for details). Also if quotas should be used, enable the quota usage during the file system creation.

For information on unified file and object access, see *Planning for unified file and object access* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

**Note:** Ensure that all GPFS file systems used to export data via NFS are mounted with the `syncnfs` option in order to prevent clients from running into data integrity issues during failover. It is recommended to use the **mmchfs** command to set the `syncnfs` option as default when mounting the GPFS file system.

For more information on creating protocol data exports, see *File system considerations for the NFS protocol* and *Fileset considerations for creating protocol data exports* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

# Configuring with the spectrumscale installation toolkit

You can use the configuration options of the **spectrumscale** installation toolkit to configure GPFS and protocols on an ongoing basis, as an alternative to the other GPFS cluster creation and configuration commands.

For detailed information about using the **spectrumscale** installation toolkit to configure GPFS and protocols, see the following:

- **spectrumscale command** in *IBM Spectrum Scale: Command and Programming Reference*
- *Installing IBM Spectrum Scale on Linux nodes and deploying protocols* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
- *Using the spectrumscale installation toolkit to perform installation tasks: Explanations and examples* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Deleting a Cluster Export Services node from an IBM Spectrum Scale cluster

Use this information to delete a Cluster Export Services (CES) node from an IBM Spectrum Scale cluster.

1. On a node other than the one you want to delete from the cluster, issue the following command to suspend the node.

   ```
   # mmces node suspend –N <Node_to_Delete>
   ```

2. On the node that you want to delete from the cluster, issue the following commands to stop the CES services.

   ```
   # mmces service stop nfs
   # mmces service stop smb
   # mmces service stop obj
   ```

   In this example, it is assumed that all three protocols are enabled on the node that you want to delete from the cluster.

3. On a node other than the one you want to delete from the cluster, issue the following command to disable CES on the node.

   ```
   # mmchnode –N <Node_to_Delete> --ces-disable
   ```

4. On a node other than the one you want to delete from the cluster, issue the following command to shut down GPFS on the node.

   ```
   # mmshutdown –N <Node_to_Delete>
   ```

5. On a node other than the one you want to delete from the cluster, issue the following command to delete the node from the cluster.

   ```
   # mmdelnode –N <Node_to_Delete>
   ```

# Chapter 3. Configuring and tuning your system for GPFS

In addition to configuring your GPFS cluster, you need to configure and tune your system.

For more information, see *GPFS cluster creation considerations* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Values suggested here reflect evaluations made at the time this documentation was written. For the latest system configuration and tuning settings, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) and the IBM Spectrum Scale Wiki (www.ibm.com/developerworks/community/wikis/home/wiki/General Parallel File System (GPFS)).

Additional GPFS and system configuration and tuning considerations include:

1. "General system configuration and tuning considerations"
2. "Linux configuration and tuning considerations" on page 32
3. "AIX configuration and tuning considerations" on page 35

For information on installing and configuring Windows on systems that will be added to a GPFS cluster, see *Configuring Windows* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

For more information on using multiple token servers, see "Using multiple token servers" on page 612

## General system configuration and tuning considerations

You need to take into account some general system configuration and tuning considerations. This topic points you to the detailed information.

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

Configuration and tuning considerations for all systems include:

1. "Clock synchronization"
2. "GPFS administration security"
3. "Cache usage" on page 30
4. Chapter 4, "Parameters for performance tuning and optimization," on page 37
5. "Access patterns" on page 32
6. "Aggregate network interfaces" on page 32
7. "Swap space" on page 32

### Clock synchronization

The clocks of all nodes in the GPFS cluster must be synchronized. If this is not done, NFS access to the data, as well as other GPFS file system operations may be disrupted.

### GPFS administration security

Before administering your GPFS file system, make certain that your system has been properly configured for security.

This includes:

- Assigning root authority to perform all GPFS administration tasks except:
  - Tasks with functions limited to listing GPFS operating characteristics.
  - Tasks related to modifying individual user file attributes.
- Establishing the authentication method between nodes in the GPFS cluster.
  - Until you set the authentication method, you cannot issue any GPFS commands.
- Designating a remote communication program for remote shell and remote file copy commands.
  - The default remote communication commands are **scp** and **ssh**. You can designate any other remote commands if they have the same syntax.
  - Regardless of which remote commands have been selected, the nodes that you plan to use for administering GPFS must be able to execute commands on any other node in the cluster without the use of a password and without producing any extraneous messages.

## Cache usage

GPFS creates a number of cache segments on each node in the cluster. The amount of cache is controlled by three attributes.

These attributes have default values at cluster creation time and may be changed through the **mmchconfig** command:

**pagepool**

The GPFS pagepool is used to cache user data and file system metadata. The pagepool mechanism allows GPFS to implement read as well as write requests asynchronously. Increasing the size of pagepool increases the amount of data or metadata that GPFS can cache without requiring synchronous I/O. The amount of memory available for GPFS pagepool on a particular node may be restricted by the operating system and other software running on the node.

The optimal size of the pagepool depends on the needs of the application and effective caching of its re-accessed data. For systems where applications access large files, reuse data, benefit from GPFS prefetching of data, or have a random I/O pattern, increasing the value for **pagepool** may prove beneficial. However, if the value is set too large, GPFS will start with the maximum that the system allows. See the GPFS log for the value it is running at.

To change the pagepool to 4 GB:

```
mmchconfig pagepool=4G
```

**maxFilesToCache**

The total number of different files that can be cached at one time. Every entry in the file cache requires some pageable memory to hold the content of the file's inode plus control data structures. This is in addition to any of the file's data and indirect blocks that might be cached in the page pool.

The total amount of memory required for inodes and control data structures can be estimated as:

```
maxFilesToCache × 3 KB
```

Valid values of **maxFilesToCache** range from 1 to 100,000,000. For systems where applications use a large number of files, of any size, increasing the value for **maxFilesToCache** may prove beneficial. This is particularly true for systems where a large number of small files are accessed. The value should be large enough to handle the number of concurrently open files plus allow caching of recently used files.

If the user does not specify a value for **maxFilesToCache**, the default value is 4000.

**maxStatCache**

This parameter sets aside additional pageable memory to cache attributes of files that are not currently in the regular file cache. This is useful to improve the performance of both the system and GPFS **stat()** calls for applications with a working set that does not fit in the regular file cache.

The memory occupied by the stat cache can be calculated as:

`maxStatCache` × `400 bytes`

Valid values of **maxStatCache** range from 0 to 100,000,000. For systems where applications test the existence of files, or the properties of files, without actually opening them (as backup applications do), increasing the value for **maxStatCache** may prove beneficial.

If the user does not specify values for **maxFilesToCache** and **maxStatCache**, the default value of **maxFilesToCache** is 4000, and the default value of **maxStatCache** is 1000.

If the user specifies a value for **maxFilesToCache** but does not specify a value for **maxStatCache**, the default value of **maxStatCache** changes to 4***maxFilesToCache**.

**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.

The total amount of memory GPFS uses to cache file data and metadata is arrived at by adding **pagepool** to the amount of memory required to hold inodes and control data structures (**maxFilesToCache** × 3 KB), and the memory for the stat cache (**maxStatCache** × 400 bytes) together. The combined amount of memory to hold inodes, control data structures, and the stat cache is limited to 50% of the physical memory on a node running GPFS.

During configuration, you can specify the **maxFilesToCache**, **maxStatCache**, and **pagepool** parameters that control how much cache is dedicated to GPFS. These values can be changed later, so experiment with larger values to find the optimum cache size that improves GPFS performance without negatively affecting other applications.

The **mmchconfig** command can be used to change the values of **maxFilesToCache**, **maxStatCache**, and **pagepool**. The **pagepool** parameter is the only one of these parameters that may be changed while the GPFS daemon is running. A **pagepool** change occurs immediately when using the **-i** option on the **mmchconfig** command. Changes to the other values are effective only after the daemon is restarted.

For further information on these cache settings for GPFS, refer to *GPFS and memory* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## The GPFS token system's effect on cache settings

Lock tokens play a role in maintaining cache consistency between nodes.

A token allows a node to cache data it has read from disk, because the data cannot be modified elsewhere without revoking the token first. Each token manager can handle approximately 300,000 different file tokens (this number depends on how many distinct byte-range tokens are used when multiple nodes access the same file). If you divide the 300,000 by the number of nodes in the GPFS cluster you get a value that should approximately equal **maxFilesToCache** (the total number of different files that can be cached at one time) + **maxStatCache** (additional pageable memory to cache file attributes that are not currently in the regular file cache).

Note the following about **maxFilesToCache** and **maxStatCache**:

- If the user does not specify values for **maxFilesToCache** and **maxStatCache**, the default value of **maxFilesToCache** is 4000, and the default value of **maxStatCache** is 1000.
- On upgrades to GPFS 4.1 from GPFS 3.4 or earlier, the existing defaults (1000 for **maxFilesToCache** and 4000 for **maxStatCache**) remain in effect.
- If the user specifies a value for **maxFilesToCache** but does not specify a value for **maxStatCache**, the default value of **maxStatCache** changes to 4***maxFilesToCache**.
- **maxFilesToCache** should be large enough to handle the number of concurrently open files plus allow caching of recently used files.
- **maxStatCache** can be set higher on user-interactive-nodes and smaller on dedicated compute-nodes, since **ls -l** performance is mostly a human response issue.

- **maxFilesToCache** and **maxStatCache** are indirectly affected by the **distributedTokenServer** configuration parameter because distributing the tokens across multiple token servers might allow keeping more tokens than if a file system has only one token server.

## Access patterns

GPFS attempts to recognize the pattern of accesses (such as strided sequential access) that an application makes to an open file. If GPFS recognizes the access pattern, it will optimize its own behavior.

For example, GPFS can recognize sequential reads and will retrieve file blocks before they are required by the application. However, in some cases GPFS does not recognize the access pattern of the application or cannot optimize its data transfers. In these situations, you may improve GPFS performance if the application explicitly discloses aspects of its access pattern to GPFS through the **gpfs_fcntl()** library call.

## Aggregate network interfaces

It is possible to aggregate multiple physical Ethernet interfaces into a single virtual interface. This is known as *Channel Bonding* on Linux and *EtherChannel/IEEE 802.3ad Link Aggregation* on AIX.

GPFS supports using such aggregate interfaces. The main benefit is increased bandwidth. The aggregated interface has the network bandwidth close to the total bandwidth of all its physical adapters. Another benefit is improved fault tolerance. If a physical adapter fails, the packets are automatically sent on the next available adapter without service disruption.

EtherChannel and IEEE802.3ad each requires support within the Ethernet switch. Refer to the product documentation for your switch to determine if EtherChannel is supported.

For details on how to configure EtherChannel and IEEE 802.3ad Link Aggregation and verify whether the adapter and the switch are operating with the correct protocols for IEEE 802.3ad, consult the operating system documentation.

**Hint:** Make certain that the switch ports are configured for **LACP** (the default is **PAGP**).

For additional service updates regarding the use of EtherChannel:

1. Go to the IBM Support Portal (www.ibm.com/support)
2. In the **Search** box, enter the search term *EtherChannel*
3. Click **Search**

**Hint:** A useful command for troubleshooting, where device is the Link Aggregation device, is:

```
entstat -d device
```

## Swap space

It is highly suggested that a sufficiently large amount of swap space is configured.

While the actual configuration decisions should be made taking into account the memory requirements of other applications, it is suggested to configure at least as much swap space as there is physical memory on a given node.

# Linux configuration and tuning considerations

Configuration and tuning considerations for the Linux nodes in your system include the use of the **updatedb** utility, the **vm.min_free_kbytes** kernel tunable, and several other options that can improve GPFS performance.

For the latest system configuration and tuning settings, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) and the IBM Spectrum Scale Wiki (www.ibm.com/developerworks/community/wikis/home/wiki/General Parallel File System (GPFS)).

For more configuration and tuning considerations for Linux nodes, see the following topics:

1. "updatedb considerations"
2. "Memory considerations"
3. "GPFS helper threads"
4. "Communications I/O" on page 34
5. "Disk I/O" on page 34

## updatedb considerations

On some Linux distributions, the system is configured by default to run the file system indexing utility **updatedb** through the **cron** daemon on a periodic basis (usually daily).

This utility traverses the file hierarchy and generates a rather extensive amount of I/O load. For this reason, it is configured by default to skip certain file system types and nonessential file systems. However, the default configuration does not prevent **updatedb** from traversing GPFS file systems. In a cluster this results in multiple instances of **updatedb** traversing the same GPFS file system simultaneously. This causes general file system activity and lock contention in proportion to the number of nodes in the cluster. On smaller clusters, this may result in a relatively short-lived spike of activity, while on larger clusters, depending on the overall system throughput capability, the period of heavy load may last longer. Usually the file system manager node will be the busiest, and GPFS would appear sluggish on all nodes. Re-configuring the system to either make **updatedb** skip all GPFS file systems or only index GPFS files on one node in the cluster is necessary to avoid this problem.

## Memory considerations

It is recommended that you adjust the **vm.min_free_kbytes** kernel tunable. This tunable controls the amount of free memory that Linux kernel keeps available (that is, not used in any kernel caches).

When **vm.min_free_kbytes** is set to its default value, on some configurations it is possible to encounter memory exhaustion symptoms when free memory should in fact be available. Setting **vm.min_free_kbytes** to a higher value (Linux **sysctl** utility could be used for this purpose), on the order of magnitude of 5-6% of the total amount of physical memory, should help to avoid such a situation.

See the GPFS Redbooks® papers for more information:

- *GPFS Sequential Input/Output Performance on IBM pSeries 690* (www.redbooks.ibm.com/redpapers/pdfs/redp3945.pdf)
- *Native GPFS Benchmarks in an Integrated p690/AIX and x335/Linux Environment* (www.redbooks.ibm.com/redpapers/pdfs/redp3962.pdf)

## GPFS helper threads

GPFS uses helper threads such as **prefetchThreads** and **workerThreads** to improve performance.

Since systems vary, it is suggested you simulate an expected workload in GPFS and examine available performance indicators on your system. For instance some SCSI drivers publish statistics in the **/proc/scsi** directory. If your disk driver statistics indicate that there are many *queued requests* it may mean you should throttle back the helper threads in GPFS.

For more information, see *Parameters for performance tuning and optimization* in *IBM Spectrum Scale: Administration Guide*.

## Communications I/O

Values suggested here reflect evaluations made at the time this documentation was written. For the latest system configuration and tuning settings, see the IBM Spectrum Scale Wiki (www.ibm.com/developerworks/community/wikis/home/wiki/General Parallel File System (GPFS)).

To optimize the performance of GPFS and your network, it is suggested you do the following:

- Enable Jumbo Frames if your switch supports it.

  If GPFS is configured to operate over Gigabit Ethernet, set the MTU size for the communication adapter to 9000.
- Verify **/proc/sys/net/ipv4/tcp_window_scaling** is enabled. It should be by default.
- Tune the TCP window settings by adding these lines to the **/etc/sysctl.conf** file:

```
# increase Linux TCP buffer limits
net.core.rmem_max = 8388608
net.core.wmem_max = 8388608
# increase default and maximum Linux TCP buffer sizes
net.ipv4.tcp_rmem = 4096 262144 8388608
net.ipv4.tcp_wmem = 4096 262144 8388608
```

  After these changes are made to the **/etc/sysctl.conf** file, apply the changes to your system:

  1. Issue the **sysctl -p /etc/sysctl.conf** command to set the kernel settings.
  2. Issue the **mmshutdown -a** command and then issue **mmstartup -a** command to restart GPFS

## Disk I/O

To optimize disk I/O performance, you should consider the following options for NSD servers or other GPFS nodes that are directly attached to a SAN over a Fibre Channel (FC) network.

1. The storage server cache settings can impact GPFS performance if not set correctly.
2. When the storage server disks are configured for RAID5, some configuration settings can affect GPFS performance. These settings include:
   - GPFS block size
   - Maximum I/O size of the Fibre Channel host bus adapter (HBA) device driver
   - Storage server RAID5 stripe size

   **Note:** For optimal performance, GPFS block size should be a multiple of the maximum I/O size of the FC HBA device driver. In addition, the maximum I/O size of the FC HBA device driver should be a multiple of the RAID5 stripe size.
3. These suggestions may avoid the performance penalty of read-modify-write at the storage server for GPFS writes. Examples of the suggested settings are:
   - 8+P RAID5
     - GPFS block size = 512K
     - Storage Server RAID5 segment size = 64K (RAID5 stripe size=512K)
     - Maximum IO size of FC HBA device driver = 512K
   - 4+P RAID5
     - GPFS block size = 256K
     - Storage Server RAID5 segment size = 64K (RAID5 stripe size = 256K)
     - Maximum IO size of FC HBA device driver = 256K

   For the example settings using 8+P and 4+P RAID5, the RAID5 parity can be calculated from the data written and will avoid reading from disk to calculate the RAID5 parity. The maximum IO size of the FC HBA device driver can be verified using **iostat** or the Storage Server performance monitor. In some cases, the device driver may need to be patched to increase the default maximum IO size.
4. The GPFS parameter **maxMBpS** can limit the maximum throughput of an NSD server or a single GPFS node that is directly attached to the SAN with a FC HBA. The default value is 2048. The

**maxMBpS** parameter is changed by issuing the **mmchconfig** command. If this value is changed, restart GPFS on the nodes, and test the read and write performance of a single node and a large number of nodes.

## AIX configuration and tuning considerations

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

## GPFS use with Oracle

When using GPFS with Oracle, configuration and tuning considerations include the following.

- When setting up your LUNs, it is important to create the NSD such that they map one to one with a LUN that is a single RAID device.
- For file systems holding large Oracle databases, set the GPFS file system block size through the **mmcrfs** command using the **-B** option, to a large value:
  - 512 KB is generally suggested.
  - 256 KB is suggested if there is activity other than Oracle using the file system and many small files exist which are not in the database.
  - 1 MB is suggested for file systems 100 TB or larger.

  The large block size makes the allocation of space for the databases manageable and has no affect on performance when Oracle is using the Asynchronous I/O (AIO) and Direct I/O (DIO) features of AIX.
- Set the GPFS worker threads through the **mmchconfig worker1Threads** command to allow the maximum parallelism of the Oracle AIO threads.
  - Adjust the GPFS prefetch threads accordingly through the **mmchconfig prefetchThreads** command. The maximum value of **prefetchThreads** plus **worker1Threads** plus **nsdMaxWorkerThreads** is 8192 on all 64-bit platforms.
  - When requiring GPFS sequential I/O, set the prefetch threads between 50 and 100 (the default is 72).

  **Note:** These changes through the **mmchconfig** command take effect upon restart of the GPFS daemon.
- The number of AIX AIO *kprocs* to create should be approximately the same as the GPFS **worker1Threads** setting.
- The AIX AIO *maxservers* setting is the number of *kprocs* PER CPU. It is suggested to set is slightly larger than the value of **worker1Threads** divided by the number of CPUs. For example if **worker1Threads** is set to 500 on a 32-way SMP, set *maxservers* to 20.
- Set the Oracle database block size equal to the LUN segment size or a multiple of the LUN pdisk segment size.
- Set the Oracle read-ahead value to prefetch one or two full GPFS blocks. For example, if your GPFS block size is 512 KB, set the Oracle blocks to either 32 or 64 16 KB blocks.
- Do not use the **dio** option on the **mount** command as this forces DIO when accessing *all* files. Oracle automatically uses DIO to open database files on GPFS.
- When running Oracle RAC 10g, it is suggested you increase the value for **OPROCD_DEFAULT_MARGIN** to at least 500 to avoid possible random reboots of nodes.

  In the control script for the Oracle CSS daemon, located in **/etc/init.cssd** the value for **OPROCD_DEFAULT_MARGIN** is set to 500 (milliseconds) on all UNIX derivatives except for AIX. For AIX this value is set to 100. From a GPFS perspective, even 500 milliseconds maybe too low in situations where node failover may take up to a minute or two to resolve. However, if during node failure the surviving node is already doing direct IO to the **oprocd** control file, it should have the necessary tokens and indirect block cached and should therefore not have to wait during failover.

# Chapter 4. Parameters for performance tuning and optimization

Use these parameters with the **mmchconfig** command for performance tuning and optimization.

## Tuning guide for frequently-changed parameters

**Autoload**
When **Autoload** is set to yes, GPFS starts automatically on the nodes that are rebooted. The rebooted nodes rejoin the cluster, the `file system automount` option is set to yes, and the file system is mounted. The default value of this parameter is no.

**Important:** Set **Autoload** to no before fixing hardware issues and performing system maintenance.

**deadlockDetectionThreshold**
When **deadlockDetectionThreshold** is set to 0, the GPFS dead-lock detection feature is disabled. The default value of this parameters is 300 seconds.

**Important:** You must enable the GPFS dead-lock detection feature to collect debug data and resolve dead lock issue in a cluster. If dead-lock events occur frequently, fix the problem instead of disabling the feature.

**defaultHelperNodes**
The nodes added to **defaultHelperNodes** run the GPFS command. Running the GPFS command on partial nodes in a cluster, such as running the mmrestripefs command on all NSD server nodes, might have a better performance. The default value of this parameter is all nodes in cluster.

**Important:** Set the `–N` option for GPFS management commands or change the value of **defaultHelperNodes** before running the GPFS management commands.

**maxFilesToCache**
The **maxFilesToCache** parameter specifies the number of files that can be cached by each node. The value of the **maxFilesToCache** parameter can be any number ranging from 1 to 100,000,000. The default value of this parameter is 4000.

The value of this parameter must be large to handle the number of concurrently open files and allow the caching of recently used files. Changing the value of **maxFilesToCache** affects the amount of memory used on the node. In a large cluster, a change in the value of **maxFilesToCache** is greatly magnified. Increasing **maxFilesToCache** in a large cluster with hundreds of nodes increases the number of tokens a token manager needs to store. Ensure that the manager node has enough memory and **tokenMemLimit** is increased when running GPFS version 4.1.1 and earlier. Therefore, increasing the value of **maxFilesToCache** on large clusters usually happens on a subset of nodes that are used as log-in nodes, SMB and NFS exporters, email servers, and other file servers. For systems on which applications use a large number of files, increasing the value of **maxFilesToCache** might be beneficial, especially where a large number of small files are accessed.

**maxBlockSize**
The value of **maxBlockSize** must be equal or larger than the maximum block size of all file systems in the local and remote clusters. Before changing the value of this parameter, ensure that the daemon on all nodes in the cluster is not functioning. The default value is 1 MB.

**maxMBpS**

The **maxMBpS** parameter indicates the maximum throughput in megabytes per second that GPFS can submit into or out of a single node. GPFS calculates from this variable how many prefetch/writebehind threads to schedule for sequential file access.

In GPFS version 3.5 and earlier, the default value is 2048. But if the node has faster interconnect, such as InfiniBand or 40GigE or multiple links) you can set the parameter to a higher value. As a general rule, try setting **maxMBpS** to twice the I/O throughput that the node can support. For example, if the node has 1 x FDR link and the GPFS configuration parameter **verbRdma** has been enabled, then the expected throughput of the node is 6000 MB/s. In this case, set **maxMBpS** to 12000.

Setting **maxMBpS** does not guarantee the desired GPFS sequential bandwidth on the node. All the layers of the GPFS stack, including the node, the network, and the storage subsystem, must be designed and tuned to meet the I/O performance requirements.

**maxStatCache**
If the value of **maxFilesToCache** is changed, the value of **maxStatCache** must also be changed. If it is not, **maxStatCache** defaults to 4 * **maxFilesToCache**. The value of the **maxStatCache** parameter can be any number ranging from 0 to 10,000,000. The default value of this parameter is 1000.

The **maxStatCache** parameter sets aside the pageable memory to cache attributes of files that are not currently in the regular file cache. This improves the performance of stat() calls for applications with a working set that does not fit in the regular file cache. For systems where applications test the existence of files, or the properties of files, without actually opening them as backup applications do, increasing the value for **maxStatCache** can be beneficial. On a system where **maxFilesToCache** is increased to a very high value, set the value to a number less than 4 * **maxFilesToCache**.

**Important:** The **maxStatCache** is ineffective on Linux inodes. Therefore, on Linux systems, the value of **maxStatCache** must be set to the default of 0, and the value of **maxFilesToCache** can be modified.

**nsdMaxWorkerThreads**
NSD server tuning. For more information about GPFS NSD server design and tuning, see NSD Server Design and Tuning.

**Pagepool**
The **Pagepool** parameter is used to change the size of the data cache on each node. The default value is either one-third of the physical memory of the node or 1G, whichever is smaller. This value applies to new installations only. On upgrades, the existing default value is maintained.

The maximum GPFS pagepool size depends on the value of the pagepoolMaxPhysMemPct parameter and the amount of physical memory on the node. Unlike local file systems that use the operating system page cache to cache file data, GPFS allocates its own cache called the pagepool. The GPFS pagepool is used to cache user file data and file system metadata. Along with file data, the pagepool supplies memory for various types of buffers such as prefetch and write behind. The default pagepool size might be sufficient for sequential IO workloads. The default pagepool size might not be sufficient for Random IO or workloads that involve a large number of small files.

In some cases, allocating 4GB, 8GB, or more memory can improve the workload performance. For database applications that use Direct IO, pagepool is not used for any user data. The main purpose in this case is for system metadata and caching the indirect blocks for the files. For NSD server, if no applications or file system manager services are running on NSD server, the pagepool is only used transiently by the NSD worker threads to gather data from client nodes and write the data to disk. The NSD server does not cache any of the data.

**readReplicaPolicy**
The **readReplicaPolicy** parameter specifies the location from which the disk must read the replicas. The valid values are default, local and fastest. The default value is default.

By default, GPFS reads the first replica even when there is no replica on the local disk. When the value of this parameter is set to local, the policy reads replicas from the local disk only if the local disk has data. For performance considerations, this is the recommended setting for FPO environments. When the value of this parameter is set to fastest, the policy reads replicas from the disk considering the fastest based on the read I/O statistics of the disk. In a system with SSD and regular disks, the value of **fastestPolicyCmpThreshold** can be set to a greater number, such as 100, to let GPFS refresh the slow disk speed statistics less frequently.

**restripeOnDiskFailure**
The **restripeOnDiskFailure** specifies if GPFS attempts to automatically recover from certain common disk failure situations. The default value of this parameter is no.

**Important:** While deploying FPO or when the HAWC feature is enabled, set the **restripeOnDiskFailure** parameter to yes.

**tiebreakerDisks**
For a small cluster with up to eight nodes that have SAN-attached disk systems, define all nodes as quorum nodes and use tiebreaker disks. With more than eight nodes, use only node quorum. While defining the tiebreaker disks, you can use the SAN-attached NSD in the file system. The default value of this parameter is null, which means no tiebreaker disk has been defined.

**unmountOnDiskFail**
The **unmountOnDiskFail** specifies how the GPFS daemon responds when a disk failure is detected. The valid values of this parameter are yes, no, and meta. The default value is no.

**Important:** Set the value of **unmountOnDiskFail** to meta for FPO deployment or when the file system metadata and data replica are more than one.

**workerThreads**
The **workerThreads** parameter controls an integrated group of variables that tune the file system performance in environments that are capable of high sequential and random read and write workloads and small file activity.

The default value of this parameter is 48. A valid value can be any number ranging from 1 to 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 adjust the following external variables with the mmchconfig command:

- *logBufferCount*
- *preFetchThreads*
- *worker3Threads*

# Chapter 5. Configuring and tuning your system for Transparent Cloud Tiering

This topic describes about the procedure for configuring and tuning your IBM Spectrum Scale node for Transparent Cloud Tiering.

## Designating the Transparent Cloud Tiering nodes

This topic describes how to designate CES nodes as Transparent Cloud Tiering node in the IBM Spectrum Scale cluster.

Before you begin, ensure that you have installed the server package RPMs on all the nodes that you want to designate as Transparent Cloud Tiering nodes. Also ensure that a node class is present.

To start working with Transparent Cloud Tiering, the administrator first needs to designate a CES node as Transparent Cloud Tiering node in the IBM Spectrum Scale cluster. Data migration to or data recall from a cloud object storage occurs in this node.

You can designate a maximum of 4 CES nodes as Transparent Cloud Tiering nodes in the IBM Spectrum Scale cluster. By default, Transparent Cloud Tiering uses the node IP addresses, not the CES IPs.

**Note:** You need to perform this procedure only on a single node where the server package is installed.

1. To designate the nodes as Transparent Cloud Tiering nodes, issue a command according to this syntax: **mmchnode change-options -N {Node[,Node...] | NodeFile | NodeClass} [--cloud-gateway-nodeclass CloudGatewayNodeClass]**.

   You can either choose to designate all nodes or only some selected nodes in a node class as Transparent Cloud Tiering nodes.

   To designate all nodes in the node class, *TCTNodeClass1*, as Transparent Cloud Tiering server nodes, issue this command:

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

   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
   ```

   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.

   **Note:** The Transparent Cloud Tiering node must have connectivity to the object storage service and access to WAN to be able to communicate with a public cloud storage provider.
2. To list the designated Transparent Cloud Tiering nodes, issue this command: **mmcloudgateway node list**

   **Note:** For more information, see the **mmcloudgateway** command.
3. To disable two nodes, *node1* and *node2*, from the node class, *TCTNodeClass1*, issue this command:

   ```
   mmchnode --cloud-gateway-disable -N nod1,node2 --cloud-gateway-nodeclass TCTNodeClass1
   ```

You can add a new node to the node class at any time. For example, issue the following commands to add the node, *10.11.12.13*, to the node class, *TCTNodeClass1*.

1. **mmchnodeclass** TCTNodeClass1 add -N 10.11.12.13
2. **mmchnode** --cloud-gateway-enable -N 10.11.12.13 --cloud-gateway-nodeclass TCTNodeClass1

## Associating a file system with the Transparent Cloud Tiering nodes

After you designate the Transparent Cloud Tiering nodes, you must associate an IBM Spectrum Scale file system with these nodes.

Before you begin, ensure that Transparent Cloud Tiering nodes are designated. For more information, see "Designating the Transparent Cloud Tiering nodes" on page 41.

To associate a file system with the Transparent Cloud Tiering nodes, issue a command according to this syntax:

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

For example, to associate the file system, */dev/gpfs0*, with the node class, *TCTNodeClass1*, issue this command:

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

To list the file system associated with the TCTNodeClass1 class, issue this command:

```
mmcloudgateway filesystem list --cloud-nodeclass TCTNodeClass1
```

**Note:**

- You need to perform this task only on any node that is designated as the Transparent Cloud Tiering node.
- All Transparent Cloud Tiering nodes are tied to a single file system. You cannot associate more than one file system with a node class.

After completing this task, you must start Transparent Cloud Tiering services on all nodes. For more information, see "Starting the Transparent Cloud Tiering services" on page 595.

## Pre-validating the cloud account settings

Before you create a cloud account, verify that the settings that you are going to use for creating a cloud storage tier are correct. This ensures that no error messages are displayed while you create a cloud storage account. This is an optional procedure.

The following parameters are validated:

- Cloud account credentials
- Cloud URL
- Cloud type (if supported or not)
- Create a test container and check if storing a test object is possible (test container is removed later)
- Other sample metrics

**Note:** Before you pre-validate the cloud account, ensure that the server RPMs are installed on the node where you run this test.

To verify the cloud account settings, issue a command according to this syntax:

```
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]
```

For example, to verify the account settings for IBM Cloud Object Storage version 3.7.2 and above with the credential of "user1", issue this command:

```
mmcloudgateway account pre-test --cloud-type cleversafe-new --username "user1" --pwd-file
 MyFile --cloud-url http://10.10.11.12
```

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
Sample data size used to calculate IOPS: 1KB
Approx. IOPS number for Puts : 16.0
Approx. IOPS number for Gets : 84.0
Sample data size used to calculate throughput: 100 MB
Approx. Data Throughput : 25 MB/s
-------------------------------------
Details:
Successful Vault operations : PUT(Vault), PUT(Object), GET(Object)
Successful Object operations: PUT, GET, HEAD, DELETE
Unsuccessful Vault operations : None
Unsuccessful Object operations : None
-------------------------------------
mmcloudgateway: Command completed.
```

**Note:** The **mmcloudgateway account pre-test** command creates a container on the cloud and is used for migrate or recall tests. After the test is run, the container is automatically deleted. However, for the IBM Cloud Object Storage cloud types, this container does not get deleted even after the test is complete, and therefore, you must manually delete it.

For more information, see the topic *mmcloudgateway command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

**Next step**: "Creating a cloud storage tier."

## Creating a cloud storage tier

This topic describes how to configure a cloud object storage account by using the Transparent Cloud Tiering node.

**Note:**

- Before you try to configure a cloud storage account, ensure that the Transparent Cloud Tiering service is started. For more information, see "Starting the Transparent Cloud Tiering services" on page 595.
- Ensure that Network Time Procotol (NTP) is enabled and time is correctly set.

You can perform this task on any Transparent Cloud Tiering node.

1. To configure a new cloud object storage account for the IBM Cloud Object Storage version 3.7.2 and above, issue a command similar to this:

   ```
   mmcloudgateway account create --cloud-nodeclass TCTNodeClass1 --cloud-name
   cscloud --cloud-type CLEVERSAFE-NEW --username  user1  --pwd-file MyFile
   --enable true --cloud-url http://10.10.10.10
   ```

   where,

   - `TCTNodeClass1` is the node class.
   - `CLEVERSAFE-NEW` is the name of the cloud object storage provider.
   - `user1` is the username.

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

- `MyFile` is the file containing the admin password.
- `TRUE` enables the added cloud object storage account for the Transparent Cloud Tiering node.
- `/ibm/scale0` is the mount point where GPFS file system is mounted on the Transparent Cloud Tiering node.
- `XYZ` is the container prefix
- `http://10.10.10.10` is the URL of the cloud object storage provider

  This is optional for Amazon "S3" but mandatory for "SWIFT," "SWIFT- KEYSTONE," and IBM Cloud Object Storage. For object storage providers that require an endpoint URL, the endpoint URL need not have a container or vault name.

2. To create a cloud account for the SWIFT-KEYSTONE cloud type, issue a command similar to this:

```
mmcloudgateway account create --cloud-nodeclass TCTNodeClass1 --cloud-name mycloud
--cloud-type SWIFT-KEYSTONE --username admin --pwd-file MyFile --enable TRUE
--cloud-url http://10.20.30.40:5050/v2.0 --tenant-id admin
```

3. 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
```

To verify that the cloud storage is created, issue this command:

```
mmcloudgateway test --cloud-nodeclass TCTNodeClass1 --cloud-name mycloud
```

These commands create a cloud storage account with some default settings. To modify the Transparent Cloud Tiering node parameters, see "Tuning Transparent Cloud Tiering parameters" on page 45.

**Note:** You can associate a file system with a single cloud storage account.

For more information on how to configure a cloud storage tier, see the **mmcloudgateway** man page.

## Enabling a cloud tiering policy

After you create a cloud storage tier, it is necessary to install a policy to migrate and recall files to and from the cloud storage.

**Note:** Failure to install the recall policy results in no data being returned when you access a file that has been migrated to the cloud.

A sample policy is located in `/usr/lpp/mmfs/samples/ilm`. This policy can be modified as needed. Ensure that the recall rules are present in the policy:

```
/* Rules for transparent recall from the cloud tier */
RULE 'OpenRead'
EVENT 'OPEN_READ'
ACTION(System('/opt/ibm/MCStore/bin/mcstore recall -c -i ' || varchar(INODE) || ' -g ' ||
varchar(GENERATION) || ' -s 0' || ' -f ' || varchar(FS_ID)) = 0)
WHERE(XATTR('dmapi.MCEA', 5, 1) == 'N')
RULE 'else' EVENT 'OPEN_READ' DIRECTORIES_PLUS

RULE 'OpenWrite'
EVENT 'OPEN_WRITE'
ACTION(System('/opt/ibm/MCStore/bin/mcstore recall -c -i ' || varchar(INODE) || ' -g ' ||
varchar(GENERATION) || ' -s 0' || ' -f ' || varchar(FS_ID)) = 0)
WHERE(XATTR('dmapi.MCEA', 5, 1) == 'N')
RULE 'else' EVENT 'OPEN_WRITE' DIRECTORIES_PLUS
```

Create a file containing the new policy with migration and recall rules, and apply the policy to the file system by using the **mmchpolicy** command.

## Tuning Transparent Cloud Tiering parameters

This topic describes the procedure for tuning Transparent Cloud Tiering parameters in IBM Spectrum Scale.

You need to change the Transparent Cloud Tiering parameters only if the default settings do not suit your requirements.

To tune Transparent Cloud Tiering parameters, issue a command according to this syntax:

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

where,

- <--cloud-nodeclass> is the node class configured for the Transparent Cloud Tiering nodes
- <--port> is the internal port that the Transparent Cloud Tiering service uses to listen for configuration commands.
- <--slice-size> is the internal unit of transferring data within the **Transparent Cloud Tiering** modules. Higher slice size indicates better performance. Default value is 256 KB.
- <--migrate-threadpool-size> 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> enables administrators to increase the parallelism for data recall 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-level> is to set non-default tracing levels for various Transparent Cloud Tiering internal components to generate more debug data if any problem occurs.
- <--audit-enable> enables the 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> 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> specifies the user name of the IBM Security Key Lifecycle Manager server REST Global Admin. Default value is SKLMAdmin.
- <--pwd-file> specifies the password file of the IBM Security Key Lifecycle Manager server REST Global Admin.

For more information on how to modify the Transparent Cloud Tiering parameters, see the **mmcloudgateway** man page.

## Integrating Transparent Cloud Tiering metrics with performance monitoring tool

This topic describes the procedure for integrating Transparent Cloud Tiering server nodes with performance monitoring tool.

Performance monitoring collector (pmcollector) and sensor (pmsensors) services are installed under `/opt/IBM/zimon`.

**Note:** You must install the sensors on all the Transparent Cloud Tiering nodes, but you need to install the collectors on any one of the GPFS nodes.

Two types of configurations are possible for the performance monitoring tool integration:

- File-based configuration (manual configuration of the sensor files as described here)
- GPFS-based configuration (configuration by using GPFS commands).

For more information on each of these methods, see *Configuring the performance monitoring tool* in *IBM Spectrum Scale: Problem Determination Guide*.

To integrate the performance monitoring tool with Transparent Cloud Tiering server nodes, do the following steps:

1. Copy `/opt/IBM/zimon/defaults/ZIMonSensors.cfg` to `/opt/IBM/zimon`. This configuration file determines which sensors are active and their properties.

   **Note:** If the sensors are already configured at `/opt/IBM/zimon/defaults/ZIMonSensors.cfg`, use the same sensors.
2. Edit the `/opt/IBM/zimon/ZIMonSensors.cfg` file and set an IP address for the “host” attribute in the “collectors" section.

   **Note:** If the collectors are already configured at `/opt/IBM/zimon/ZIMonSensors.cfg`, use the same collectors.
3. Edit the `/opt/IBM/zimon/ZIMonSensors.cfg` file to append the following Transparent Cloud Tiering sensors at the end of the sensor configuration section:

   ```
   {
   # Transparent Cloud Tiering statistics
   name = "MCStoreGPFSStats"
   period = 10
   type = "Generic"
   },
   {
   # Transparent Cloud Tiering statistics
   name = "MCStoreIcstoreStats"
   period = 10
   type = "Generic"
   },
   {
   # Transparent Cloud Tiering statistics
   name = "MCStoreLWEStats"
   period = 10
   type = "Generic"
   }
   ```

   **Note:** Each sensor should be separated by a comma. The period is the frequency in seconds at which the performance monitoring tool polls the Transparent Cloud Tiering service for statistics. The period is set to 10 seconds but it is a configurable value. The sensor is turned off when the period is set to 0.
4. Copy the following files from `/opt/ibm/MCStore/config` folder to `/opt/IBM/zimon` folder.

- MCStoreGPFSStats.cfg
- MCStoreIcstoreStats.cfg
- MCStoreLWEStats.cfg

5. Restart the sensors by using this command: **service pmsensors restart** .
6. Restart the collectors by using these commands: **service pmcollector restart**

   **Note:**

   If the collector is already installed and is running, then only **pmsensors** service needs to be restarted. If you are installing both **pmcollectors** and **pmsensors**, then both services need to be restarted.

## Integrating Transparent Cloud Tiering metrics with performance monitoring tool by using GPFS-based configurations

This topic describes the procedure for integrating Transparent Cloud Tiering metrics with performance monitoring tool by using GPFS-based configuration.

1. Copy the following files from `/opt/ibm/MCStore/config` folder to `/opt/IBM/zimon` folder:
   - MCStoreGPFSStats.cfg
   - MCStoreIcstoreStats.cfg
   - MCStoreLWEStats.cfg

   **Note:** It works the same as in the file-based configuration and has to be located in the `/opt/IBM/zimon/` directory.
2. Register the sensor in the GPFS configuration by storing the following snippet in the `MCStore-sensor-definition.cfg` file:

```
sensors=
{
    # Transparent Cloud Tiering statistics
    name = "MCStoreGPFSStats"
    period = 10
    type = "Generic"
   },

   {
     # Transparent Cloud Tiering statistics
     name = "MCStoreIcstoreStats"
     period = 10
     type = "Generic"
   },

  {
    # Transparent Cloud Tiering statistics
    name = "MCStoreLWEStats"
    period = 10
    type = "Generic"
  }
```

3. Run this command:

```
prompt# mmperfmon config add --sensors MCStore-sensor-definition.cfg
```

   **Note:** The sensor definition file can list multiple sensors separated by commas (,).

For more information on GPFS-based configuration, see the topic *mmperfmon command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

## Enabling Transparent Cloud Tiering performance monitoring metrics on the GUI

This topic describes the procedure for enabling performance metrics on the IBM Spectrum Scale GUI.

To enable performance metrics on the GUI, do the following steps:

1. Edit the `/usr/lpp/mmfs/gui/conf/gpfsgui.properties` file and set the ENABLE_MCSTORE property value to "true".

   **Note:** This file needs to be modified on each GUI node.
2. Issue this command: **systemctl restart gpfsgui**

## Configuring Transparent Cloud Tiering with SKLM

You need to perform certain configurations to be able to use the security features of IBM Spectrum Scale.

**Note:** Before you configure SKLM, ensure that the following steps are completed:

1. SKLM server is installed. For more information on SKLM, see "Preparation for encryption" on page 543.
2. A cloud storage account is created. For more information, see "Creating a cloud storage tier" on page 43.

To configure Transparent Cloud Tiering with IBM Security Key Lifecycle Manager (SKLM), do the following steps:

1. Configure SKLM by issuing a command according to the following syntax. The command is all on one line:

   ```
   mmcloudgateway config set --cloud-nodeclass <CloudNodeClass> --rkm-enable TRUE --rkm-servername <hostname>
   --rkm-port 9080 --rkm-username <SKLMAdmin> --pwd-file <PasswordFile> --force
   ```

   where
   - `<CloudNodeClass>` is the node class that was created with the **mmcrnodeclass** command.
   - `<hostname>` is the host name of the SKLM server
   - `<SKLMAdmin>` is the admin user name of the SKLM server
   - `<PasswordFile>` is the file containing the password. It points to a temporary file location that contains the administrator password.

     **Note:**
     - If no password file is given, the system prompts for a password.
     - If the **--force** option is not specified, the command displays an error message that ensures that you recall all the migrated data before switching to Remote Key Manager.
2. The command above adds the following properties to the `.mcstore_settings` file:

   ```
   rkm.keybackend0.host=<SKLM HOST>
   keymanager=rkm
   rkm.keybackends=keybackend0
   rkm.keybackend0.provider=com.ibm.icstore.keymanager.keybackend.SKLMKeyBackend
   rkm.keybackend0.port=<SKLM PORT>
   rkm.keybackend0.devicegroup=MCSTORE
   rkm.keybackend0.protocols=TLSv1.2
   rkm.keybackend0.ciphersuites=TLS_RSA_WITH_AES_128_CBC_SHA,TLS_RSA_WITH_AES_256
   _CBC_SHA256,TLS_RSA_WITH_AES_128_CBC_SHA256
   rkm.keybackend0.fips=off
   ```

```
rkm.keybackend0.storefile=<file_system_location>/.mcstore/clientCred.p12
rkm.keybackend0.storepasswd=<PASSWORD>
rkm.keybackend0.storetype=pkcs12
encpolicy0.keys=<SKLM KEY>\:rkm.keybackend0
```

3. Restart the Transparent Cloud Tiering service by issuing a command according to this syntax:

   ```
   mmcloudgateway service start -N <NodeClass>
   ```

   where <NodeClass> is the node class for which the service is started.

To see the UUID and device group of the current active SKLM encryption key, enter the following command:

```
mmcloudgateway config list --cloud-nodeclass <CloudNodeClass>
```

where <CloudNodeClass> is the node class that was created with the **mmcrnodeclass** command.

# Chapter 6. Configuring Active File Management

The following topics list the parameters required to configure Active File Management.

## Configuration parameters for AFM

The following table lists the AFM and AFM-based DR configuration parameters with their default values and the commands to set and edit the parameters.

*Table 6. Configuration parameters at cache and their default values at the cache cluster*

| AFM configuration parameter | Unit | Description | Mode on AFM |
|---|---|---|---|
| **afmAsyncDelay** | seconds | Indicates the time at which the requests start flushing to the home cluster. For write-intensive applications that write 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 the data on the remote cluster. | SW, IW |
| **afmDirLookupRefreshInterval** | seconds | Defines the frequency of revalidation triggered by a look-up operation such as ls or stat on a directory from the cache cluster. AFM sends a message to the home cluster to find out whether the metadata of that directory is modified since it was last revalidated. If so, the latest metadata information at the home cluster is reflected on the cache cluster. | RO, LU, IW |
| **afmDirOpenRefreshInterval** | seconds | Defines the frequency of revalidations triggered by the read and update operations on a directory from the cache cluster. AFM sends a message to the home cluster to find whether the metadata of that directory is modified since it was last revalidated. Open requests on files or sub-directories on that directory are served from the cache fileset until the **afmDirOpenRefreshInterval** expires, after which the open requests are sent to the home cluster. | RO, LU, IW |
| **afmDisconnectTimeout** | seconds | Defines the interval until which the MDS waits after it detects that the home cluster is inaccessible before declaring the outage by moving the cache cluster state to Disconnected. | RO, SW, IW, LU |

*Table 6. Configuration parameters at cache and their default values at the cache cluster (continued)*

| AFM configuration parameter | Unit | Description | Mode on AFM |
|---|---|---|---|
| **afmExpirationTimeout** | seconds | Is used with **afmDisconnectTimeout** to control the duration of a network outage between the cache and home clusters before the data in the cache expires and becomes unavailable until a home reconnection occurs. | RO |
| **afmFileLookupRefreshInterval** | seconds | Defines the frequency of revalidation triggered by a look-up operation on a file such as ls or stat, from the cache cluster. AFM sends a message to the home cluster to determine that the metadata of the file is modified since it was last revalidated. If so, the latest metadata information at home is reflected on the cache cluster. | RO, LU, IW |
| **afmFileOpenRefreshInterval** | seconds | Defines the frequency of revalidations triggered by the read and write operations on a file from the cache cluster. AFM sends a message to the home cluster to determine that the metadata of the file is modified since it was last revalidated.<br><br>Open requests on the file are served from the cache fileset until the **afmFileOpenRefreshInterval** expires, after which the open requests are sent to the home cluster. | RO, LU, IW |
| **afmEnableAutoEviction** | | Indicates if automatic eviction is triggered on a fileset. | RO, SW, IW,LU |
| **afmPrefetchThreshold** | Whole number - 0 to 100 | Controls the partial file caching feature.<br><br>0: Full file prefetching after reading 3 blocks<br><br>1-99: The percentage of the file size that must be cached before the entire file is pulled into the cache cluster<br>100: Disables full file prefetching. Only fetches and caches data blocks that are read by the application. When all data blocks have been cached, the file is marked as cached. | RO, SW, IW,LU |
| **afmShowHomeSnapshot** | | Controls the visibility of the home snapshot directory in the cache cluster. For this to be visible in the cache cluster, this variable must to be set to yes, and the snapshot directory name in the cache and home clusters must not be the same. | RO, LU |

*Table 6. Configuration parameters at cache and their default values at the cache cluster (continued)*

| AFM configuration parameter | Unit | Description | Mode on AFM |
|---|---|---|---|
| **afmReadSparseThreshold** | Bytes | When a sparse file at the home cluster is read into the cache, the cache cluster maintains the sparseness, if the size of the file exceeds the **afmReadSparseThreshold**. If the size of a file is less than the threshold, sparseness is not maintained at the cache cluster. | RO, SW, IW,LU |
| **afmHashVersion** | | Specifies the older or newer version of the gateway node hashing algorithm, such as, the **mmchconfig afmHashVersion=2**, to minimize the impact of gateway nodes joining or leaving the active cluster by running as few recoveries as possible. | Not applicable |
| **afmMountRetryInterval** | seconds | Specifies the interval after which the MDS retries an operation at home, in cases where home is in an unhealthy state (see Unmounted, Dropped states). Needs the GW node recycle or `-i` option for immediate effect. | RO, SW, IW,LU |

*Table 7. Configuration parameters at cache and their default values at the cache cluster - Valid values*

| AFM configuration parameter | Valid values | Default value | Tunable at the cluster level | Tunable at the fileset level |
|---|---|---|---|---|
| **afmAsyncDelay** | 1 - 2147483647 | 15 | Yes | Yes |
| **afmDirLookupRefreshInterval** | 0 - 2147483647 | 60 | Yes | Yes |
| **afmDirOpenRefreshInterval** | 0 - 2147483647 | 60 | Yes | Yes |
| **afmDisconnectTimeout** | 0 - 2147483647, disable | 60 | Yes | No |
| **afmExpirationTimeout** | 0 - 2147483647, disable | disabled | Yes | Yes |
| **afmFileLookupRefreshInterval** | 0 - 2147483647 | 30 | Yes | Yes |
| **afmFileOpenRefreshInterval** | 0 - 2147483647 | 30 | Yes | Yes |
| **afmEnableAutoEviction** | Yes \| No | Yes | No | Yes |
| **afmPrefetchThreshold** | 0 - 100 | 0 | No | Yes |
| **afmShowHomeSnapshot** | Yes \| No | No | Yes | Yes |
| **afmReadSparseThreshold** | 0 - 2147483647 | 128 M | Yes | Yes |
| **afmHashVersion** | 1 \| 2 | 1 | Yes | No |
| **afmMountRetryInterval** | | 300 | No | No |

## Parallel I/O configuration parameters for AFM

The parameters in the following table can be used at the cache cluster for tuning parallel I/O. All parallel I/O parameters do not need the gateway node daemon recycle for the new value to take effect.

*Table 8. Configuration parameters at cache for parallel I/O*

| AFM configuration parameter | Unit | Description | Mode on AFM |
|---|---|---|---|
| **afmNumFlushThreads** | Whole number | 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. Ensure that you do not set this parameter to a very high value. | SW, IW |
| **afmNumReadThreads** | Whole number | Defines the number of threads used on each participating gateway node during a parallel read. The default value of this parameter is 1. That is, one reader thread is active on every gateway node for each big read operation qualifying for splitting as per the parallel read threshold value. | SW, IW,RO, LU |
| **afmNumWriteThreads** | Whole number | Defines the number of threads used on each participating gateway node during a parallel write. The default value of this parameter is 1. That is, one writer thread is active on every gateway node for each big write operation qualifying for splitting as per the parallel write threshold value. | SW, IW |
| **afmParallelReadChunkSize** | MB | Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel reads. Values are in bytes. | SW, IW, RO, LU |
| **afmParallelReadThreshold** | MB | Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when the file size exceeds this threshold value. Values are in MB. The default value is 1024 MB. | SW, IW, RO, LU |
| **afmParallelWriteChunkSize** | MB | Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel writes. Values are in bytes. | SW, IW |
| **afmParallelWriteThreshold** | MB | Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when file size exceeds this threshold value. Values are in MB. The default value is 1024 MB. | SW, IW |

Table 8. Configuration parameters at cache for parallel I/O (continued)

| AFM configuration parameter | Unit | Description | Mode on AFM |
|---|---|---|---|
| **afmHardMemThreshold** | MB | Sets 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 the Dropped state. Exceeding the limit can occur if the cache cluster is disconnected for an extended time or if the connection with the home cluster has low bandwidth and a lot pending requests are accumulated in the queue. After the value of this parameter is changed, a gateway node daemon recycle is required for the new value to take effect. | |

Table 9. Configuration parameters at cache for parallel I/O - valid values

| AFM configuration parameter | Valid values | Default value | Tunable at the cluster level | Tunable at the fileset level |
|---|---|---|---|---|
| **afmNumFlushThreads** | 1 - 1024 | 4 | No | Yes |
| **afmNumReadThreads** | 1 - 64 | 1 | Yes | Yes |
| **afmNumWriteThreads** | 1 - 64 | 1 | Yes | Yes |
| **afmParallelReadChunkSize** | 0 - 2147483647 | 128 | Yes | Yes |
| **afmParallelReadThreshold** | 0 - 2147483647 | 1024 | Yes | Yes |
| **afmParallelWriteChunkSize** | 0 - 2147483647 | 128 | Yes | Yes |
| **afmParallelWriteThreshold** | 0 - 2147483647 | 1024 | Yes | Yes |
| **afmHardMemThreshold** | | 5 G | Yes | No |

# Chapter 7. Configuring AFM-based DR

The following topics list the parameters required to configure AFM-based DR.

## Configuration parameters for AFM-based DR

The following table lists the AFM-based DR configuration parameters with their default values and the commands to set and edit the parameters:

*Table 10. Configuration parameters at primary and their default values*

| AFM configuration parameter | Unit | Description | Mode on AFM-DR |
|---|---|---|---|
| **afmAsyncDelay** | Seconds | Indicates the time at which the requests start flushing to the home cluster. For write-intensive applications that write 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 the data on the remote cluster. | primary |
| **afmDisconnectTimeout** | Seconds | Defines the interval until which the MDS waits after it detects that the home cluster is inaccessible before declaring the outage by moving the cache cluster state to Disconnected. | primary |
| **afmReadSparseThreshold** | Bytes | When a sparse file at the home cluster is read into the cache, the cache cluster maintains the sparseness, if the size of the file exceeds **afmReadSparseThreshold**. If the size of a file is less than the threshold, sparseness is not maintained at the cache cluster. | primary |
| **mountRetryInterval** | Seconds | Specifies the interval after which the MDS retries an operation at home, in cases where the home is in an unhealthy state. Needs the GW node recycle or `-i` option for immediate effect. It is tunable per gateway. The updated value is visible only as long as the gateway is active and is applicable for all filesets owned by that gateway. | primary |
| **afmRPO** | Minutes | Specifies the interval for generating RPO snapshots. | primary |

*Table 11. Configuration parameters at primary and their default values - Valid values*

| AFM configuration parameter | Valid values | Default value | Tunable at the cluster level | Tunable at the fileset level |
|---|---|---|---|---|
| **afmAsyncDelay** | 1 - 2147483647 | 15 | Yes | Yes |

Table 11. Configuration parameters at primary and their default values - Valid values (continued)

| AFM configuration parameter | Valid values | Default value | Tunable at the cluster level | Tunable at the fileset level |
|---|---|---|---|---|
| **afmDisconnectTimeout** | 0 - 2147483647, disable | 60 | Yes | No |
| **afmReadSparseThreshold** | 0 - 2147483647 | 128 | Yes | Yes |
| **mountRetryInterval** | | 300 | No | No |
| **afmRPO** | 5 - 2147483647 | 15 | No | Yes |

## Parallel I/O configuration parameters for AFM-based DR

All parallel I/O parameters do not need the gateway node daemon recycle for the new value to take effect.

The parameters in the following table can be used at the cache cluster for tuning parallel I/O:

Table 12. Configuration parameters at cache for parallel I/O

| AFM configuration parameter | Unit | Description | Mode on AFM DR |
|---|---|---|---|
| **afmNumFlushThreads** | Whole number | 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. Do not set this parameter to an extreme value. | primary |
| **afmNumWriteThreads** | Whole number | Defines the number of threads used on each participating gateway node during a parallel write. The default value of this parameter is 1. That is, one writer thread is active on every gateway node for each big write operation qualifying for splitting as per the parallel write threshold value. | primary |
| **afmParallelWriteChunkSize** | MB | Defines the minimum chunk size of the read that needs to be distributed among the gateway nodes during parallel writes. Values are in bytes. | primary |
| **afmParallelWriteThreshold** | MB | Defines the threshold beyond which parallel reads become effective. Reads are split into chunks when file size exceeds this threshold value. Values are in MB. The default value is 1024 MB. | primary |
| **afmHardMemThreshold** | MB | Sets 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 the Dropped state. Exceeding the limit can occur if the cache cluster is disconnected for an extended time or if the connection with the home cluster has low bandwidth and therefore a lot of pending requests are accumulated in the queue. After the value of this parameter is changed, a gateway node daemon recycle is required for the new value to take effect. | |

*Table 13. Configuration parameters at cache for parallel I/O - valid values*

| AFM configuration parameter | Valid values | Default value | Tunable at the cluster level | Tunable at the fileset level |
|---|---|---|---|---|
| **afmNumFlushThreads** | 1 - 1024 | 4 | No | Yes |
| **afmNumWriteThreads** | 1 - 64 | 1 | Yes | Yes |
| **afmParallelWriteChunkSize** | 0 - 2147483647 | 128 | Yes | Yes |
| **afmParallelWriteThreshold** | 0 - 2147483647 | 1024 | Yes | Yes |
| **afmHardMemThreshold** | | 5 G | Yes | No |

# Chapter 8. Tuning for Kernel NFS backend on AFM and AFM DR

If AFM communication use the NFSv3 protocol, for peak performance, tune the gateway NFS servers that host home exports and the gateway servers that support cache/primary filesets.

Most of these tuning parameters require at least the AFM client to be restarted. Ensure that the NFS server is not mounted. Unlink the AFM fileset, or stop GPFS or IBM Spectrum Scale on the gateway node.

Tuning for 1GigE networks is different from tuning 10GigE networks. For 10GbE, all settings need to be scaled up, but not necessarily by a factor of 10. Many of these settings are affected after a server reboot. Therefore, each time a server restarts, the settings must be reset. The TCP buffer tuning is required for all 10GigE links and for 1GigE links where the value of **RTT** is greater than 0.

## Tuning the gateway node on the NFS client

You can set the maximum number of (TCP) RPC requests that can be in flight by using the **sunrpc.tcp_slot_table_entries** or **/proc/sys/sunrpc/tcp_slot_table_entries** parameter.

For 1 GigE, if there is no round-trip time leave the default value. You can increase the value to a number greater than 16 if the round-trip time is large. For 10 GigE, ensure that this value is 48 or a number greater than 48 depending on the round-trip time.

When you set the **seqDiscardThreshold** parameter, it affects AFM or AFM DR as follows:

- If I/O requests are from a node that is not the gateway node, there is no effect.
- If the read request is on the gateway node for an uncached file, a higher **seqDiscardThreshold** value results in better performance as it allows the gateway to cache more data. When the data is returned to the application, there is a greater chance that it comes out of the cache/primary cluster.

## Tuning on both the NFS client (gateway) and the NFS server (the home/secondary cluster)

This topic describes the tuning on both the NFS client (gateway) and the NFS server (the home/secondary cluster).

You must set TCP values that are appropriate for the delay (`buffer size = bandwidth * RTT`).

For example, if your ping time is 50 ms, and the end-to-end network consists of all 100BT Ethernet and OC3 (155 Mbps), the TCP buffers must be the following: `0.05 sec * 10 MB/sec = 500 KB`

If you are connected using a T1 line (1 Mbps) or less, do not change the default buffers. Faster networks usually benefit from buffer tuning.

The following parameters can also be used for tuning. A 12194304 buffer size is provided here as an example value for a 1 GigE link with a delay of 120ms. To set these values, set the following configurations in a file and load it with `sysctl -p` filename.

The following are example values. Initial testing is required to determine the best value for a particular system:

```
net.ipv4.tcp_rmem = 12194304 12194304 12194304
          net.ipv4.tcp_wmem = 12194304 12194304 12194304
          net.ipv4.tcp_mem = 16777216 16777216 16777216
          net.core.rmem_max = 12194304
          net.core.wmem_max = 12194304
          net.core.rmem_default = 12194304
          net.core.wmem_default = 12194304
          net.core.optmem_max = 12194304
          net.core.netdev_max_backlog = 250000
          net.ipv4.tcp_no_metrics_save = 1
          net.ipv4.tcp_timestamps = 0
          net.ipv4.tcp_sack = 1
```

**Note:** For TCP tuning, the **sysctl** value changes do not take effect until a new TCP connection is created, which occurs at NFS mount time. Therefore, for TCP changes, it is critical that the AFM fileset and the NFS client are unmounted and GPFS is shut down.

With Red Hat Enterprise Linux 6.1 and later, both the NFS client and the server perform TCP auto-tuning. It automatically increases the size of the TCP buffer within the specified limits through **sysctl**. If the client or the server TCP limits are too low, the TCP buffer grows for various round-trip time between the GPFS clusters. With Red Hat Enterprise Linux 6.1 and earlier, NFS is limited in its ability to tune the TCP connection. Therefore, do not use a version earlier than Red Hat Enterprise Linux 6.1 in the cache/primary cluster.

As a GPFS cluster might be handling local and remote NFS clients, you can set the GPFS server values for the largest expected round-trip time of any NFS client. This ensures that the GPFS server can handle clients at various locations. Then on the NFS clients, set the TCP buffer values that are appropriate for the SONAS cluster that they are accessing.

The gateway node is both an NFS server for standard NFS clients if they exist and an NFS client for communication with the home/secondary cluster. Ensure that the TCP values are set appropriately, because values that are either too high or too low can negatively impact performance.

If performance continues to be an issue, increase the buffer value by at the most 50%. An increase in size that is greater than 50% has a negative effect on the performance.

## Tuning the NFS server on the home/secondary cluster or the NFS server

This topic describes the tuning on the NFS server on the home/secondary cluster or the NFS server.

*Table 14. NFS server parameters*

| Parameter name with path | Description |
|---|---|
| /proc/fs/nfsd/max_block_size | Set to 1 MB for improved performance. |
| /proc/fs/nfsd/threads | Set to a minimum value of 32. You can set this value to larger than 128 depending on the throughput capacity and the round-trip time between the cache/primary and home/secondary clusters. Determining the correct value might take a few trials. |

*Table 14. NFS server parameters (continued)*

| Parameter name with path | Description |
|---|---|
| `nfsPrefetchStrategy` | Set it to a number between 5 and 10. As AFM uses NFS, ensure that this is set on the home/secondary GPFS cluster.<br><br>After the NFS values are set, you can mount and access the AFM filesets. The first time the fileset is accessed the AFM NFS client mounts the home/secondary server or servers. To see these mounts on a gateway node, enter the following command: **`cat /proc/mounts`**.<br><br>The system displays the mount point and the mount options. If the **`wsize`** and **`rsize`** values are not 1 MB, you can adjust the parameters and mount the AFM filesets again to get the correct values. |

# Chapter 9. Performing GPFS administration tasks

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

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

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

The information for administration and maintenance of GPFS and your file systems is covered in topics including:

1. "Requirements for administering a GPFS file system" and "Common GPFS command principles" on page 67
2. Chapter 1, "Configuring the GPFS cluster," on page 1
3. Chapter 10, "Managing file systems," on page 71
4. Chapter 12, "Managing disks," on page 113
5. Chapter 17, "Managing GPFS quotas," on page 227
6. Chapter 19, "Managing GPFS access control lists," on page 245
7. **Command reference** in *IBM Spectrum Scale: Command and Programming Reference*
8. **GPFS programming interfaces** in *IBM Spectrum Scale: Command and Programming Reference*
9. **GPFS user exits** in *IBM Spectrum Scale: Command and Programming Reference*
10. Chapter 20, "Considerations for GPFS applications," on page 275
11. Chapter 11, "File system format changes between versions of GPFS," on page 109

## Requirements for administering a GPFS file system

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

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

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

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

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

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

You can avoid configuring your GPFS nodes to allow remote access to the root user ID, by using sudo wrapper scripts to run GPFS administrative commands. See "Running IBM Spectrum Scale without remote root login" on page 16.

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

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

## adminMode configuration attribute

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

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

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

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

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

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

**Note List:**

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

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

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

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

```
mmchconfig adminMode=central
```

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

## Common GPFS command principles

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

Those principles include:

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

## Specifying nodes as input to GPFS commands

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

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

*Node*
: A representation of an individual node, which can be any of these:
  - Short GPFS administration node interface name.
  - Long GPFS administration node interface name.
  - Short GPFS daemon node interface name.
  - Long GPFS daemon node interface name.
  - IP address corresponding to the GPFS daemon node interface.
  - GPFS node number.

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

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

  **all**
  : All of the nodes in the GPFS cluster.

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

**localhost**
The node on which the command is running.

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

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

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

**nsdnodes**
All of the NSD server nodes in the GPFS cluster.

**quorumnodes**
All of the quorum nodes in the GPFS cluster.

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

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

For commands operating on a file system, the stripe group manager node is always implicitly included in the node list. Not every GPFS command supports all of the node specification options described in this topic. To learn what kinds of node specifications are supported by a particular GPFS command, see the relevant command description in *Command reference* in *IBM Spectrum Scale: Command and Programming Reference*.

## Stanza files

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

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

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

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

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

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

GPFS recognizes a number of stanzas:

**%nsd:**
NSD stanza

**%pdisk:**
Physical disk stanza

**%vdisk:**
Virtual disk stanza

**%da:**
Declustered array stanza

**%rg:**
Recovery group stanza

The details are documented under the corresponding commands.

For more information about the IBM Spectrum Scale RAID commands that use stanzas, see *IBM Spectrum Scale RAID: Administration* in Elastic Storage Server (ESS) documentation on IBM Knowledge Center.

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

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

Here is what a stanza file may look like:

```
# Sample file containing two NSD stanzas

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

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

**Note:** The server name used in the NSD stanza file must be resolvable by the system.

## Listing active GPFS commands

You can list the active GPFS commands that are running on the file system manager node.

Most GPFS commands run within the GPFS daemon on the file system manager node. Even if you start a command on another node of the cluster, the node typically sends the command to the file system manager node to be executed. (Two exceptions are the **mmdiag** command and the **mmfsadm dump** command, which run on the node where they were started.)

To list the active commands on the file system manager node, follow these steps:

1. Enter the **mmlsmgr** command with no parameters to discover which node is the file system manager node. For more information on other options available for the **mmlsmgr** command, see the topic *mmlsmgr command* in the *IBM Spectrum Scale: Command and Programming Reference* guide. In the following example, the **mmlsmgr** command reports that `node05` is the file system manager node:

```
# mmlsmgr
file system      manager node
---------------- ------------------
gpfs1            192.168.145.14 (node05)

Cluster manager node: 192.168.145.13 (node03)
```

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

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

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

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

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

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

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

The remaining input is detailed debugging data that is used for product support. For more information on **mmdiag** command output, see the topic *mmdiag command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

# Chapter 10. Managing file systems

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

For information on how to create GPFS file systems, see the *A sample file system creation* section in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the *mmcrfs command*.

File system management tasks include:

1. "Mounting a file system"
2. "Unmounting a file system" on page 73
3. "Deleting a file system" on page 74
4. "Determining which nodes have a file system mounted" on page 74
5. "Checking and repairing a file system" on page 75
6. "Listing file system attributes" on page 77
7. "Modifying file system attributes" on page 78
8. "Querying and changing file replication attributes" on page 79
9. "Using Direct I/O on a file in a GPFS file system" on page 80
10. "Restriping a GPFS file system" on page 88
11. "Querying file system space" on page 89
12. "Querying and reducing file system fragmentation" on page 90
13. "Protecting data in a file system using backup" on page 92
14. "Scale Out Backup and Restore (SOBAR)" on page 99

Managing filesets, storage pools and policies is also a file system management task. For more information on managing storage pools, filesets and policies, see Chapter 22, "Information lifecycle management for IBM Spectrum Scale," on page 293. Use the following information to manage file systems in IBM Spectrum Scale.

### Managing file system through GPFS GUI

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

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

## Mounting a file system

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

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

To mount a GPFS file system, enter:

```
mmmount device
```

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

```
mmmount fs1
```

## Mounting a file system on multiple nodes

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

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

```
mmmount fs1 -a
```

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

## GPFS-specific mount options

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

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

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

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

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

The GPFS-specific mount options are:

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

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

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

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

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

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

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

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

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

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

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

**syncnfs**
Synchronously commit metadata changes coming from the NFS daemon.

**useNSDserver={always | asfound | asneeded | never}**
Controls the initial disk discovery and failover semantics for NSD disks. The possible values are:

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

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

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

**never**
Always use local disk access.

## Changing a file system mount point on protocol nodes

If required, you can change a file system mount point on IBM Spectrum Scale protocol nodes.

To change a file system mount point on protocol nodes, perform the following steps:

1. Unmount the file system:
   ```
   mmumount fs0 -a
   ```
2. Change the mount point:
   ```
   mmchfs fs0 -T /ibm/new_fs0
   ```
3. Change the path of all NFS and SMB exports.

   **Note:** The **mmnfs export change** and the **mmsmb export change** commands do not allow path names to be edited. Therefore, path names need to be removed and added back in.
4. Change the object CCR files:
   - account-server.conf
   - container-server.conf
   - object-server-.conf
   - object-server-sof.conf
   - spectrum-scale-object.comf
   - spectrum-scale-objectizer.conf

   The parameter that you need to change varies depending on the configuration file.
   a. Use the **mmobj config change** command to list the parameters for the file. For example, to list the parameters for the object-server.conf file, enter:
      ```
      mobj config list --ccrfile object-server.conf --section DEFAULT --property devices
      ```
   b. Use the **mmobj config change --ccrfile** *file name* to change the parameter. For example, to change the object-server.conf file, enter:
      ```
      mmobj config change --ccrfile object-server.conf --section DEFAULT --property devices
       /newFS/name
      ```

## Unmounting a file system

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

If the file system does not unmount, see the *File system fails to unmount* section in the *IBM Spectrum Scale: Problem Determination Guide*.

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

```
mmumount device
```

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

```
mmumount fs1
```

## Unmounting a file system on multiple nodes

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

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

```
mmumount fs1 -a
```

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

---

## Deleting a file system

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

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

```
mmdelfs fs1
```

The system displays information similar to:

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

    gpfs9nsd
    gpfs13nsd
    gpfs11nsd
    gpfs12nsd

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

For more information, see the following:

- "Unmounting a file system" on page 73
- **mmdelfs command** in *IBM Spectrum Scale: Command and Programming Reference* for complete usage information
- **mmdelnsd command** in *IBM Spectrum Scale: Command and Programming Reference* for removing the NSD definitions after deleting the file system

---

## Determining which nodes have a file system mounted

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

Note that the **mmlsmount -L** command reports file systems that are in use at the time the command is issued. A file system is considered to be in use if it is explicitly mounted with the **mount** or **mmmount** command or if it is mounted internally for the purposes of running some other GPFS command. For example, when you run the **mmrestripefs** command, the file system will be internally mounted for the

duration of the command. If **mmlsmount** is issued in the interim, the file system will be reported as being in use by the **mmlsmount** command but, unless it is explicitly mounted, will not show up in the output of the **mount** or **df** commands. For more details, see For more information, see the topic *The mmlsmount command* in the *IBM Spectrum Scale: Problem Determination Guide*..

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

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

## Checking and repairing a file system

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

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

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

**Note:**

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

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

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

- Various problems related to filesets: missing or corrupted fileset metadata, inconsistencies in directory structure related to filesets, missing or corrupted fileset root directory, other problems in internal data structures. The repaired filesets will be renamed as **Fileset***FilesetId* and put into unlinked state.

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

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

```
gpfs_fsck

<header>
  sgid = "C0A87ADC:5555C87F"
  disk_data_version = 1
  fs_name = "gpfsh0"
  #patch_file_version = 1
  #start_time = "Fri May 15 16:32:58 2015"
  #fs_manager_node = "h0"
  #fsck_flags = 150994957
</header>

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

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

  <patch_dir>
    entry_offset = 48
    entry_fold_value = 306661480
    delete_entry = true
  </patch_dir>
</patch_block>

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

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

<patch_inode>
```

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

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

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

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

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

```
mmfsck fs1
```

For complete usage information, see **mmchdisk command**, **mmcheckquota command**, **mmfsck command**, and **mmlsdisk command** in *IBM Spectrum Scale: Command and Programming Reference*

## Dynamic validation of descriptors on disk

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

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

## Listing file system attributes

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

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

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

```
mmlsfs gpfs1
```

The system displays information similar to:

```
 flag                value                    description
------------------- ------------------------ -----------------------------------
 -f                 8192                     Minimum fragment size in bytes
 -i                 4096                     Inode size in bytes
 -I                 16384                    Indirect block size in bytes
 -m                 2                        Default number of metadata replicas
 -M                 2                        Maximum number of metadata replicas
 -r                 2                        Default number of data replicas
```

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

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

For complete usage information, see **mmlsfs command** in *IBM Spectrum Scale: Command and Programming Reference*. For a detailed discussion of file system attributes, see *GPFS architecture* and *File system creation considerations* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Modifying file system attributes

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

**Note:** All files created after issuing the **mmchfs** command take on the new attributes. Existing files are not affected. Use the **mmchattr** or **mmrestripefs -R** command to change the replication factor of existing files. See "Querying and changing file replication attributes" on page 79.

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

```
mmchfs fs1 -r 2
```

To confirm the changes, enter:

```
mmlsfs fs1 -r
```

The system displays information similar to:

```
flag               value                    description
------------------- ------------------------ -----------------------------------
 -r                2                        Default number of data replicas
```

For complete usage information, see **mmchfs command** and **mmlsfs command** in *IBM Spectrum Scale: Command and Programming Reference*. For a detailed discussion of file system attributes, see *GPFS architecture* and *File system creation considerations* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Querying and changing file replication attributes

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

For complete usage information, see **mmlsattr command** and **mmchattr command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Querying file replication

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

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

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

The system displays information similar to:

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

See the **mmlsattr command** in *IBM Spectrum Scale: Command and Programming Reference* for complete usage information. For a detailed discussion of file system attributes, see *GPFS architecture* and *File system creation considerations* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Changing file replication attributes

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

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

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

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

To confirm the change, enter:

```
mmlsattr /fs1/project7.resource
```

The system displays information similar to:

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

See the **mmchattr command** and the **mmlsattr command** in *IBM Spectrum Scale: Command and Programming Reference* for complete usage information. For a detailed discussion of file system attributes, see *GPFS architecture* and *File system creation considerations* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

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

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

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

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

## File compression

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

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

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



### Comparison with object compression

File compression and object compression use the same compression technology but are available in different environments and are configured in different ways. Object compression is available in the Cluster Export Systems (CES) environment and is configured with the **mmobj policy** command. With object compression, you can create an object storage policy that periodically compresses new objects and files in a GPFS fileset.

File compression is available in non-CES environments and is configured with the **mmapplypolicy** command or directly with the **mmchattr** command.

## When to use file compression

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

## Setting up file compression and decompression

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

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

**Note:** File compression and decompression with the **mmapplypolicy** command is not supported on Windows.
With the **mmchattr** command, you specify the **--compression** option and the names of the files or filesets that you want to compress or decompress. For example, the following command compresses a file:

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

The following command decompresses the same file:

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

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

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

```
RULE 'COMPR1' MIGRATE FROM POOL 'datapool' COMPRESS('yes') WHERE NAME LIKE 'green%'
```

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

```
RULE 'COMPR1' MIGRATE FROM POOL 'datapool' COMPRESS('no') WHERE NAME LIKE 'green%'
```

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

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

For more information, see the following help topics:

- The topic *mmchattr command* in the *IBM Spectrum Scale: Command and Programming Reference*
- "Overview of policies" on page 300
- "Policy rules: Syntax" on page 302
- "Policy rules: Terms" on page 303

When you do file compression, you can defer the compression operation a later time. For more information, see the subtopic "Deferred file compression" on page 82.

## Warning

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

- Doing file compression or decompression. This includes compression or decompression with the **mmchattr** command or with a policy and the **mmapplypolicy** command.
- Running the **mmrestripefile** command or the **mmrestripefs**, either to complete a deferred file compression or decompression, or for any other reason.

## Reported size of compressed files

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

## Deferred file compression

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

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

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

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

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

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

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

## Indicators of file compression or decompression

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

**COMPRESSED**

The **mmlsattr** command displays the **COMPRESSED** indicator on the `Misc attributes` line of its output. See the example of **mmlsattr** output in Figure 1 on page 83. If present, **COMPRESSED** indicates that the file is compressed or is marked for deferred compression. If absent, the absence indicates that the file is uncompressed or is marked for deferred decompression.

**Note:**

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

**illCompressed**

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

**Note:**

- This indicator reflects the state of the GPFS_IAFLAG_ILLCOMPRESSED flag in the **gpfs_iattr64_t** structure of the inode of the file. For more information about this structure, see the topic *gpfs_iattr64_t_structure* in the *IBM Spectrum Scale: Command and Programming Reference*.
- Some file system events can cause the **illCompressed** flag to be set. Consider the following examples:
  - When data is written into an already compressed file, the existing data remains compressed but the new data is uncompressed. The **illCompressed** flag is set for this file.
  - When a compressed file is memory-mapped, the memory-mapped area of the file is decompressed before it is read into memory. The **illCompressed** flag is set for this file.

  For more information, see the subtopic "Updates to compressed files" on page 84.

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

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

*Figure 1. Compression and decompression indicators*

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

*Table 15. COMPRESSED and illCompressed indicators*

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

## Partially compressed files

The **COMPRESSED** flag is set when the user selects the file to be compressed through the **mmchattr --compress yes** command or a policy run. The flag indicates that the user wants the file to be compressed.

If the user specifies the **-I defer** command option with the **mmchattr** command or policy run, the **illCompressed** flag is set during the command execution or the policy run. The file's **illCompressed** flag indicates that the request to compress the file has not been fulfilled. The **illCompressed** flag is reset at the conclusion of the actual compression execution on the file, after **mmrestripefs -z** or **mmrestripefile -z** command finishes compressing the file if the **-I defer** option was used. The **illCompressed** flag can be set again upon contents updates on the file that cause update-driven uncompression.

The compressibility of a file can change over time if its contents are changed. Different parts of a file may have different compressibility. Based on the 10% space-saving criterion (see the subtopic "Limitations" on page 85), some compression groups (in granularity of 10 data blocks) of a file might be compressed while others are not.

In sum, the state of the **Compressed** flag, on or off, indicates the intention of the user to compress the file or not. The **illCompressed** flag indicates the compression execution status. The actual compression status of the data blocks depends on the **illCompressed** and **Compressed** flags as well as the compressibility of the current data.

## Updates to compressed files

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

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

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

For more information, see the preceding subtopic "Deferred file compression" on page 82.

## File compression and memory mapping

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

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

## File compression and direct I/O

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

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

## Backing up and restoring compressed files

Files are decompressed when they are moved out of storage that is directly managed by IBM Spectrum Scale. This fact affects file backups by products like IBM Spectrum Protect, IBM Spectrum Protect for Space Management (HSM), IBM Spectrum Archive™, Transparent Cloud Tiering (TCT), and others. When

you back up a file with these products, the file system decompresses the file data inline when it is read by the backup agent. The file system also sets the **illCompressed** flag in the file properties. The backed-up file data is not compressed.

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

### FPO environment

File compression supports a File Placement Optimizer (FPO) environment or horizontal storage pools.

**FPO block group factor:** Before you compress files in an File Placement Optimizer (FPO) environment, you must set the block group factor to a multiple of 10. If you do not, then data block locality is not preserved and performance is slower.

For compatibility reasons, before you do file compression with an AFM cache or FPO files, you must upgrade the whole cluster to version 4.2.1 or later. To verify that the cluster is upgraded, follow these steps:

1. At the command line, enter the **mmlsconfig** command with no parameters.
2. In the output, verify that `minReleaseLevel` is >= 4.2.1.0.

## Limitations

Notice the restrictions stated in the preceding subtopics:

- "File compression and memory mapping" on page 84
- "File compression and direct I/O" on page 84
- "Backing up and restoring compressed files" on page 84

File compression has the following limitations:

- File compression in this release is designed to be used only for compressing cold data or write-once objects and files. Compressing other types of data can result in performance degradation. File compression uses the zlib data compression library and favors saving space over speed.
- File compression processes each compression group within a file independently. A compression group consists of one to ten consecutive data blocks within a file. If the file contains fewer than ten data blocks, the whole file is one compression group. If the space savings for a compression group is less than 10%, file compression does not compress it but skips to the next compression group.
- For file-enabled compression in an FPO-enabled file system, the block group factor must be a multiple of 10 so that the compressed data maintains data locality. If the block group factor is not a multiple of 10, the data locality is broken.
- Direct I/O is not supported for compressed files.
- The following operations are not supported:
  - Compressing files in snapshots
  - Compressing a clone
  - Compressing files in an AFM cache site or in an AFM-based asynchronous Disaster Recovery (DR) fileset.
  - Compressing small files (files that consume fewer than two subblocks, compressing small files into an inode).
  - Compressing files other than regular files, such as directories.
  - Cloning a compressed file
- On Windows:

– Compression or decompression with the **mmapplypolicy** command is not supported.
– Compression of files in Windows hyper allocation mode is not supported.
– The following Windows APIs are not supported:
  - FSCTL_SET_COMPRESSION to enable/disable compression on a file
  - FSCTL_GET_COMPRESSION to retrieve compression status of a file
– In Windows Explorer, in the Advanced Attributes window, the compression feature is not supported.

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

QoS limits the effect of I/O-intensive GPFS maintenance commands on overall system I/O performance.

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

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

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

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

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

For more information, see the descriptions of the QoS commands:
- *mmchqos command* in the *IBM Spectrum Scale: Command and Programming Reference*
- *mmlsqos command* in the *IBM Spectrum Scale: Command and Programming Reference*

**Note:**
- QoS requires the file system to be at V4.2.0.0 or later. To check the file system level, enter the following command:

  `mmlsfs` *`fileSystemName`* `-V`
- QoS works with asynchronous I/O, memory-mapped I/O, cached I/O, and buffered I/O. However, with direct I/O, QoS counts the IOPS but does not regulate them.

## Overview of using QoS

The following steps provide an overview of how to use QoS. In this overview, assume that the file system `fs0` contains 5 nodes and has two storage pools: the system storage pool (`system`) and another storage pool `sp1`.

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

*Table 16. Set QoS classes to* **unlimited**

| Storage pool | QoS class: maintenance | QoS class: other |
|---|---|---|
| `system` | **unlimited** | **unlimited** |

*Table 16. Set QoS classes to* **unlimited** *(continued)*

| Storage pool | QoS class: maintenance | QoS class: other |
|---|---|---|
| sp1 | **unlimited** | **unlimited** |

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

b. Run some maintenance commands that drive I/O on all nodes and disks.

c. Run the **mmlsqos** command to observe how many IOPS are consumed:

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

2. Run the **mmchqos** command to allocate the available IOPS among the storage pools.

a. Allocate a smaller share of IOPS to the **maintenance** class, perhaps 15 percent. For example, if you determined in Step 1 that the maximum is 10,000 IOPS, then you might allocate 1500 IOPS to the maintenance class.

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

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

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

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

*Table 17. Allocate the available IOPS*

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

The command is on one line:

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

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

```
mmdeldisk fs0 nsd12
```

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

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

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

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

```
mmchqos fs0 --disable
```

When you reenable QoS it starts applying the allocations again:

```
mmchqos fs0 --enable
```

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

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

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

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

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

### See also

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

## Restriping a GPFS file system

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

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

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

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

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

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

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

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

1. Use the **mmchdisk** command to suspend any disks to which you *do not* want the file system restriped. You may want to exclude disks from file system restriping because they are failing. See "Changing GPFS disk states and parameters" on page 119.
2. Use the **mmlsdisk** command to assure that all disk devices to which you *do* want the file system restriped are in the up/normal state. See "Displaying GPFS disk states" on page 118.

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

```
mmrestripefs fs1 -b
```

The system displays information similar to:

```
Scanning file system metadata, phase 1 ...
  19 % complete on Wed Mar 14 21:28:46 2012
 100 % complete on Wed Mar 14 21:28:48 2012
Scan completed successfully.
Scanning file system metadata, phase 2 ...
Scanning file system metadata for sp1 storage pool
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
 100.00 % complete on Wed Mar 14 21:28:55 2012
Scan completed successfully.
```

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

For complete usage information, see **mmrestripefs command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Querying file system space

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

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

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

```
mmdf fs1
```

The system displays information similar to:

```
disk                disk size  failure holds    holds              free KB             free KB
name                    in KB    group metadata data        in full blocks        in fragments
--------------- ------------- -------- -------- ----- -------------------- -------------------
Disks in storage pool: system (Maximum disk size allowed is 122 GB)
hd16vsdn10           17793024       -1 yes      yes        17538560 ( 99%)          1728 ( 0%)
hd3vsdn01             8880128        2 yes      yes         8658176 ( 98%)          1600 ( 0%)
hd4vsdn01             8880128        2 yes      yes         8616448 ( 97%)          1384 ( 0%)
hd15vsdn10           17793024       10 yes      yes        17539584 ( 99%)          1664 ( 0%)
hd13vsdn02            8880128     4001 yes      yes         8663552 ( 98%)          1776 ( 0%)
```

```
hd8vsdn01             8880128     4002 yes      yes          8659200 ( 98%)          1936 ( 0%)
hd5vsdn01             8880128     4002 yes      yes          8654848 ( 97%)          1728 ( 0%)
hd33n09              17796008     4003 yes      yes         17540864 ( 99%)          2240 ( 0%)
                -------------                           -------------------- -------------------
(pool total)        257800488                               252091136 ( 98%)         46928 ( 0%)

Disks in storage pool: fs1sp1 (Maximum disk size allowed is 122 GB)
hd30n01               8897968        8 no       yes          8895488 (100%)           424 ( 0%)
hd31n01               8897968        8 no       yes          8895488 (100%)           424 ( 0%)
                -------------                           -------------------- -------------------
(pool total)         17795936                                17790976 (100%)           848 ( 0%)

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

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

For complete usage information, see **mmdf command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Querying and reducing file system fragmentation

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

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

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

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

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

## Querying file system fragmentation

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

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

```
mmdefragfs fs0 -i
```

The system displays information similar to:

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

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

For complete usage information, see **mmdefragfs command** in *IBM Spectrum Scale: Command and Programming Reference*.

## Reducing file system fragmentation

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

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

```
mmdefragfs fs1 -u 100
```

The system displays information similar to:

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

Defragmenting until full block utilization is 98.00%, currently 97.07%
 27.35 % complete on Tue May 26 14:25:42 2009    (    617882 inodes       4749 MB)
 82.65 % complete on Tue May 26 14:26:02 2009    (   1867101 inodes      10499 MB)
 89.56 % complete on Tue May 26 14:26:23 2009    (   2023206 inodes      14296 MB)
 90.01 % complete on Tue May 26 14:26:43 2009    (   2033337 inodes      17309 MB)
 90.28 % complete on Tue May 26 14:27:03 2009    (   2039551 inodes      19779 MB)
 91.17 % complete on Tue May 26 14:27:23 2009    (   2059629 inodes      23480 MB)
 91.67 % complete on Tue May 26 14:27:43 2009    (   2070865 inodes      26760 MB)
 92.51 % complete on Tue May 26 14:28:03 2009    (   2089804 inodes      29769 MB)
 93.12 % complete on Tue May 26 14:28:23 2009    (   2103697 inodes      32649 MB)
 93.39 % complete on Tue May 26 14:28:43 2009    (   2109629 inodes      34934 MB)
 95.47 % complete on Tue May 26 14:29:04 2009    (   2156805 inodes      36576 MB)
 95.66 % complete on Tue May 26 14:29:24 2009    (   2160915 inodes      38705 MB)
 95.84 % complete on Tue May 26 14:29:44 2009    (   2165146 inodes      40248 MB)
 96.58 % complete on Tue May 26 14:30:04 2009    (   2181719 inodes      41733 MB)
 96.77 % complete on Tue May 26 14:30:24 2009    (   2186053 inodes      43022 MB)
 96.99 % complete on Tue May 26 14:30:44 2009    (   2190955 inodes      43051 MB)
 97.20 % complete on Tue May 26 14:31:04 2009    (   2195726 inodes      43077 MB)
 97.40 % complete on Tue May 26 14:31:24 2009    (   2200378 inodes      43109 MB)
 97.62 % complete on Tue May 26 14:31:44 2009    (   2205201 inodes      43295 MB)
 97.83 % complete on Tue May 26 14:32:05 2009    (   2210003 inodes      43329 MB)
 97.85 % complete on Tue May 26 14:32:25 2009    (   2214741 inodes      43528 MB)
 97.86 % complete on Tue May 26 14:32:55 2009    (   2221888 inodes      43798 MB)
 97.87 % complete on Tue May 26 14:33:35 2009    (   2231453 inodes      44264 MB)
 97.88 % complete on Tue May 26 14:34:26 2009    (   2243181 inodes      45288 MB)
 100.00 % complete on Tue May 26 14:35:10 2009

               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 the **mmdefragfs command** in *IBM Spectrum Scale: Command and Programming Reference* for complete usage information.

---

## Protecting data in a file system using backup

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

You can use the **mmbackup** command to back up the files of a GPFS file system or the files of an independent fileset to an IBM Spectrum Protect server.

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

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

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

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

The **mmbackup** command can be used to back up some or all of the files of a GPFS file system to IBM Spectrum Protect servers using the IBM Spectrum Protect Backup-Archive client. After files have been backed up, you can restore them using the interfaces provided by IBM Spectrum Protect.

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

The **mmbackup** command can interoperate with regular IBM Spectrum Protect commands for backup and expire operations. However if after using **mmbackup**, any IBM Spectrum Protect incremental or selective backup or expire commands are used, **mmbackup** needs to be informed of these activities. Use either the **-q** option or the **--rebuild** option in the next **mmbackup** command invocation to enable **mmbackup** to rebuild its shadow databases. (See **mmbackup Examples** in *IBM Spectrum Scale: Command and Programming Reference*.)

These databases *shadow* the inventory of objects in IBM Spectrum Protect so that only new changes will be backed up in the next incremental **mmbackup**. Failing to do so will needlessly back up some files additional times. The shadow database can also become out of date if **mmbackup** fails due to certain IBM

Spectrum Protect server problems that prevent **mmbackup** from properly updating its shadow database after a backup. In these cases it is also required to issue the next **mmbackup** command with either the **-q** option or the **--rebuild** options.

The **mmbackup** command provides:

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

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

The **mmbackup** command supports backing up GPFS file system data to multiple IBM Spectrum Protect servers. The ability to partition file backups across multiple IBM Spectrum Protect servers is particularly useful for installations that have a large number of files. For information on setting up multiple IBM Spectrum Protect servers, see "IBM Spectrum Protect requirements" on page 94.

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

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

If you have file systems that were backed up using the GPFS 3.2 or earlier version of the **mmbackup** command, you will not be able to take advantage of some of the new **mmbackup** features until a new full backup is performed. See "File systems backed up using GPFS 3.2 or earlier versions of mmbackup" on page 95.

## Protecting data in a fileset using the `mmbackup` command

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

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

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

## IBM Spectrum Protect requirements

The **mmbackup** command requires a IBM Spectrum Protect client and server environment to perform a backup operation.

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

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

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

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

   A very large include-exclude list can decrease backup performance. Use wildcards and eliminate unnecessary include statements to keep the list as short as possible.
6. If more than one node will be used to perform the backup operation (**mmbackup -N** option):
   - The **mmbackup** command will verify that the IBM Spectrum Protect Backup-Archive client versions and configuration are correct before executing the backup. Any nodes that are not configured correctly will be removed from the backup operation. Ensure that IBM Spectrum Protect clients are installed and at the same version on all nodes that will invoke the **mmbackup** command or participate in parallel backup operations.
   - Ensure that IBM Spectrum Protect is aware that the various IBM Spectrum Protect clients are all working on the same file system, not different file systems having the same name on different client machines. This is accomplished by using proxy nodes for multiple nodes in the cluster. See the IBM Spectrum Protect documentation for recommended settings for GPFS cluster nodes setup.
7. Restoration of backed-up data must be done using IBM Spectrum Protect interfaces. This can be done with the client command-line interface or the IBM Spectrum Protect web client. The IBM Spectrum Protect web client interface must be made operational if you wish to use this interface for restoring data to the file system from the IBM Spectrum Protect server.
8. When more than one IBM Spectrum Protect server is referenced in the **dsm.sys** file, **mmbackup** uses all listed IBM Spectrum Protect servers by default. To use only a select IBM Spectrum Protect server or the servers that are listed in **dsm.sys**, use the **mmbackup --tsm-servers** option. When more than one IBM Spectrum Protect server is used for backup, the list and the order specified should remain constant. If additional IBM Spectrum Protect servers are added to the backup later, add them to the end of the list that is specified with the **mmbackup --tsm-servers** option.

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

**Attention:** If you are using the IBM Spectrum Protect Backup-Archive client command line or web interface to do back up, use caution when you unlink filesets that contain data backed up by IBM Spectrum Protect. IBM Spectrum Protect tracks files by path name 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.

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

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

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

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

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

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

## Migrating to mmbackup from IBM Spectrum Protect-interface–based backup

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

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

The **mmbackup** command uses one or more shadow database files to determine changes in the file system. To convert from the IBM Spectrum Protect interface backup to **mmbackup**, one must create the shadow database file or files by using the **--rebuild** option of **mmbackup**. The rebuild option queries the

existing IBM Spectrum Protect server or servers and creates a shadow database of the files currently backed up in IBM Spectrum Protect. After the shadow database file or files are generated, **mmbackup** can be used for all future incremental or full backups.

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

## Tuning backups with the `mmbackup` command

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

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

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

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

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

  **Tip:** Set this parameter to the number of CPU *cores* implemented on a typical node in your GPFS cluster.
- The **-n** *DirThreadLevel* parameter allows specification of the number of threads that will be created and dispatched within each **mmapplypolicy** process during the directory scan phase.

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

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

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

- Automatic computation of ideal backup bunch count. The number of objects named in each file list can be determined, separately from the number in an expire list, and automatically computed, if not specified by the user.
- As an alternative to the automatic computation, the user can control backup processing as follows:

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

For more information on the **mmbackup** tuning parameters, see **mmbackup command** in *IBM Spectrum Scale: Command and Programming Reference*.

## MMBACKUP_PROGRESS_CALLOUT environment variable

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

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

- The message class provided with this message is 0.

  Or
- At least **$progressInterval** seconds has elapsed.

  Or
- The **$progressContent** mask has a bit set which matches a bit set in the message class provided with this message.

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

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

Where:

**JOB**
Specifies the literal backup string to identify this component.

**FS** Specifies the file system device name.

**SERVER**
Specifies the IBM Spectrum Protect server currently used for backup.

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

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

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

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

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

**FILESBACKEDUP**
Specifies the number of files that were backed up during this backup job. This value remains 0 until phase **backing up** is reached, and then it increases until **dsmc** finishes. This value increases while **dsmc selective** jobs are running, and is calculated by IBM Spectrum Protect output. If the backup job fails before completion, some output may indicate files backed up but not counted. This value always increases.

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

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

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

## Backing up a file system using the GPFS policy engine

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

This process typically includes:

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

For more information on GPFS policies and rules refer to Chapter 22, “Information lifecycle management for IBM Spectrum Scale,” on page 293.

## Backing up file system configuration information

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

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

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

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

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

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

## Using APIs to develop backup applications

You can develop backup applications using APIs

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

A typical use of these subroutines is the following scenario:

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

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

---

## Scale Out Backup and Restore (SOBAR)

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

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

These commands cannot be run from a Windows node.

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

## Scheduling backups using IBM Spectrum Protect scheduler

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

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

For the following steps, these example values are assumed:

```
client-node-proxyname (asnodename)              => proxy-cluster1
Node to be used for the schedule (aka nodename) => gpfs-node1
tsm server name                                 => tsm1
file system to be backed up                     => gpfs0
global snapshot name (created for backup job)   => BKUPsnap
schedule name on the TSM server                 => proxy-cluster1_sched
```

1. On the IBM Spectrum Protect server, define the schedule using the following command.

   ```
   define schedule standard proxy-cluster1_sched type=client action=command objects=/usr/bin/
   my-mmbackup-script.sh starttime=05:00:00 startdate=today
   ```

2. On the IBM Spectrum Protect server, associate the schedule with the IBM Spectrum Scale proxy node using the following command.

   ```
   define association standard proxy-cluster1_sched proxy-cluster1
   ```

3. Create the backup script on the IBM Spectrum Scale node.

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

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

4. On one of the IBM Spectrum Protect client nodes, verify the schedule using the following command.

   ```
   dsmc q sched
   ```

## Configuration reference for using IBM Spectrum Protect with IBM Spectrum Scale

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

**Note:** Refer to the latest IBM Spectrum Protect documentation on IBM Knowledge Center for the latest information on the mentioned settings.

## Options in the IBM Spectrum Protect configuration file dsm.sys

This topic describes the options in the IBM Spectrum Protect configuration file dsm.sys.

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

## Exclude or include options

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

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

Do not add exclude statements for snapshots as snapshots are specially handled automatically by **mmbackup** and IBM Spectrum Protect options when needed.

**mmbackup** excludes the following folders from the scan by default and these need not be explicitly excluded in the `dsm.sys` file or on the IBM Spectrum Protect server:

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

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

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

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

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

   **Note:** Do not add include statements after exclude statements. Do not add exclude statements before include statements.
2. Before starting the **mmbackup** job, set the following environment variable:

   ```
   export MMBACKUP_IGNORE_INCLUDE=1
   ```

**Note:**

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

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

### Usage of a IBM Spectrum Protect proxy node (asnodename option)

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

### Important IBM Spectrum Protect client configuration option

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

## Options in the IBM Spectrum Protect configuration file dsm.opt

This topic describes the options in the IBM Spectrum Protect configuration file dsm.opt.

### Special character handling

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

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

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

### Important IBM Spectrum Protect client configuration options

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

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

## Base IBM Spectrum Protect client configuration files for IBM Spectrum Scale usage

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

**Important:** While the IBM Spectrum Protect client configuration file dsm.sys can contain node specific information, it cannot simply be copied from node to node without touching or correcting the corresponding node specific information.

The following are example contents of IBM Spectrum Protect configuration files.

**Contents of dsm.sys**

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

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

**Example values of variables used in dsm.sys**

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

**Contents of dsm.opt**

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

**Contents of dsm.opt when IBM Spectrum Protect for Space Management is used**

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

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

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

Ensure the following before you begin:

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

For information on how to create and maintain snapshots, see Chapter 23, “Creating and maintaining snapshots of file systems,” on page 347

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

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

   *device* is the name of the file system.

   ```
   # mmlssnapshot fs1

   Snapshots in file system fs1:
   Directory        SnapId  Status   Created                   Fileset
   fileset_test1    1       Valid    Mon Mar 23 09:20:37 2015  nfs-ganesha
   filesystem_test2 2       Valid    Mon Mar 23 11:12:59 2015
   ```

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

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

   ```
   # mmsnapdir fs1

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

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

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

   ```
   # mmlsfs fs1 -T

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

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

   For example:

   ```
   Full path to the file:  /gpfs/fs1/nfs-ganesha/test1/
   Default mount point:    /gpfs/fs1
   Truncated path:         /nfs-ganesha/test1/
   ```

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

   The full snapshot path is:

   *filesystem_default_mountpoint/snapshot_directory/snapshot_name/truncated_path*

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

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

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

     **cp -p** *full_snapshot_path/file_name restore_directory*
   - If you want to restore a directory, change the directory to the *restore_directory* and use the following command:

     **tar -zcf** *tar_file_name full_snapshot_path/directory_name*

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

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

Ensure the following before you begin:

- You have the full path to the files or directories that you want to restore. The path must include the file system to which these files or directories belong.
- You know which snapshot contains the files or directories that you want to restore.

- You have created a restore directory to which these files or directories are to be restored to avoid accidentally overwriting files or directories.

For information on how to create and maintain snapshots, see Chapter 23, "Creating and maintaining snapshots of file systems," on page 347

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

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

   *device* is the name of the file system.

   ```
   # mmlssnapshot fs1

   Snapshots in file system fs1:
   Directory        SnapId  Status   Created                   Fileset
   fileset_test1    1       Valid    Mon Mar 23 09:20:37 2015  nfs-ganesha
   filesystem_test2 2       Valid    Mon Mar 23 11:12:59 2015
   ```

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

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

   ```
   # mmsnapdir fs1

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

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

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

   ```
   # mmlsfileset fs1

   Filesets in file system 'fs1':
   Name          Status   Path
   root          Linked   /gpfs/fs1
   nfs-ganesha   Linked   /gpfs/fs1/nfs-ganesha
   nfs-ganesha2  Linked   /gpfs/fs1/nfs-ganesha2
   nfs-ganesha3  Linked   /gpfs/fs1/nfs-ganesha3
   nfs-ganesha4  Linked   /gpfs/fs1/nfs-ganesha4
   ```

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

   For example:

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

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

   The full snapshot path is:

   *fileset_path*`/.snapshots/`*snapshot_directory*`/`*snapshot_name*`/`*truncated_path*

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

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

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

     **cp -p** *full_snapshot_path*`/`*file_name restore_directory*
   - If you want to restore a directory, change the directory to the *restore_directory* and use the following command:

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

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

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

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

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

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

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

   Where:

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

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

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

   All copies of the specified file are listed as follows:

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

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

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

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

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

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

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

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

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

# Chapter 11. File system format changes between versions of GPFS

Every GPFS file system has a format version number associated with it. This version number corresponds to the on-disk data structures of the file system and is an indicator of the supported file system functionality.

The file system version number is assigned when the file system is first created, and is updated to the latest supported level after the file system is migrated using the **mmchfs -V** command.

The format version number for a file system can be displayed with the **mmlsfs -V** command. If a file system was created with an older GPFS release, new functionality that requires different on-disk data structures will not be enabled until you run the **mmchfs -V** command. In addition to **mmchfs -V**, certain new features may require you to additionally run the **mmmigratefs** command.

**Note:** The **-V** option cannot be used to make file systems created prior to GPFS 3.2.1.5 available to Windows nodes. Windows nodes can mount only file systems that are created with GPFS 3.2.1.5 or later.

The **mmchfs -V** option requires the specification of one of two values - **full** or **compat**:

- Specifying **mmchfs -V full** enables all of the new functionality that requires different on-disk data structures. After this command, nodes in remote clusters running an older GPFS version will no longer be able to mount the file system.

  Running the **mmchfs -V full** gives you a warning similar to the following:

  **Note:**

  ```
  # mmchfs n03NsdOnFile36 -V full
  You have requested that the file system be upgraded to
  version 15.01 (4.2.0.0). This will enable new functionality but will
  prevent you from using the file system with earlier releases of GPFS.
  Do you want to continue?
  ```

- Specifying **mmchfs -V compat** enables only features that are backward compatible with nodes running GPFS 3.2. After this command, nodes in remote clusters running GPFS 3.2 or later will still be able to mount the file system, but nodes running GPFS versions 3.1 or older will not be able to mount the file system.

The current highest file system format version is 15.01. This is the version that is assigned to file systems created with IBM Spectrum Scale 4.2.0.0. The same version number will be assigned to older file systems after you run the **mmchfs -V full** command.

- After running **mmchfs -V full**, the file system will be able to support the following:
  - Quality of Service (QoS) function is enabled
  - Compression

If your current file system is at format level 14.20 (IBM Spectrum Scale 4.1.1), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
  - Enabling and disabling of quota management without unmounting the file system.
  - The use of fileset-level integrated archive manager (IAM) modes.
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 14.04 (GPFS 4.1.0.0), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support different block allocation map types on an individual storage-pool basis.
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 13.23 (GPFS 3.5.0.7), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
  - directory block sizes up to 256 KB (previous maximum was 32 KB)
  - directories will be able to reduce their size when files are removed
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 13.01 (GPFS 3.5.0.1), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
  - extended storage pool properties
  - File Placement Optimizer (FPO)
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 12.03 (GPFS 3.4), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
  - independent filesets and snapshots of individual independent filesets
  - active file management (AFM)
  - storing the data for small files in the inode
  - file clones (writable snapshots of a file)
  - policy language support for new attributes, variable names, and functions: OPTS clause for the SET POOL and RESTORE rules, encoding of path names via an ESCAPE clause for the EXTERNAL LIST and EXTERNAL POOL rules, GetEnv(), GetMMconfig(), SetXattr(), REGEX().
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 11.03 (GPFS 3.3), the set of enabled features depends on the value specified with the **mmchfs -V** option:

- After running **mmchfs -V full**, the file system will be able to support the following:
  - more than 2,147,483,648 files
  - fast extended attributes (which requires **mmmigratefs** to be run also)
- There are no new features that can be enabled with **mmchfs -V compat**.

If your current file system is at format level 10.00 (GPFS 3.2.0.0) or 10.01 (GPFS 3.2.1.5), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus the following:

- new maximum number of filesets in a file system (10000)
- new maximum for the number of hard links per object (2**32)
- improved quota performance for systems with large number of users
- policy language support for new attributes, variable names, and functions: MODE, INODE, NLINK, RDEVICE_ID, DEVICE_ID, BLOCKSIZE, GENERATION, XATTR(), ATTR_INTEGER(), and XATTR_FLOAT()

If your current file system is at format level 9.03 (GPFS 3.1), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- fine grain directory locking
- LIMIT clause on placement policies

If your current file system is at format level 8.00 (GPFS 2.3), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- storage pools
- filesets
- fileset quotas

If your current file system is at format level 7.00 (GPFS 2.2), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus:

- NFS V4 access control lists
- new format for the internal allocation summary files

If your current file system is at format level 6.00 (GPFS 2.1), after running **mmchfs -V**, the file system will be able to support all of the features included with earlier levels, plus extended access control list entries (**-rwxc** access mode bits).

The functionality described in this topic is only a subset of the functional changes introduced with the different GPFS releases. Functional changes that do not require changing the on-disk data structures are not listed here. Such changes are either immediately available when the new level of code is installed, or require running the **mmchconfig release=LATEST** command. For a complete list, see the "Summary of changes" on page xv.

# Chapter 12. Managing disks

Use the following information to manage disks in IBM Spectrum Scale

Disks can have connectivity to each node in the cluster, be managed by network shared disk servers, or a combination of the two. For more information, see *mmcrnsd command* in the *IBM Spectrum Scale: Command and Programming Reference*. Also see, *Network Shared Disk (NSD) creation considerations* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

**Note:** A LUN provided by a storage subsystem is a disk for the purposes of this documentation, even if the LUN is made up of multiple physical disks.

The disk related tasks performed on a GPFS file system include:

1. "Displaying disks in a GPFS cluster"
2. "Adding disks to a file system" on page 114
3. "Deleting disks from a file system" on page 114
4. "Replacing disks in a GPFS file system" on page 116
5. "Additional considerations for managing disks" on page 118
6. "Displaying GPFS disk states" on page 118
7. "Changing GPFS disk states and parameters" on page 119
8. "Changing your NSD configuration" on page 121
9. "Changing NSD server usage and failback" on page 122
10. "Enabling and disabling Persistent Reserve" on page 122

## Displaying disks in a GPFS cluster

You can display the disks that belong to your GPFS cluster by issuing the **mmlsnsd** command.

The default is to display information for all disks defined to the cluster (**-a**). Otherwise, you may choose to display the information for a particular file system (**-f**) or for all disks which do not belong to any file system (**-F**).

To display the default information for all of the NSDs belonging to the cluster, enter:

```
mmlsnsd
```

The system displays information similar to:

```
File system  Disk name   NSD servers
------------------------------------------------------------------------
fs2          hd3n97      c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd4n97      c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd5n98      c5n98g.ppd.pok.ibm.com,c5n97g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          hd6n98      c5n98g.ppd.pok.ibm.com,c5n97g.ppd.pok.ibm.com,c5n99g.ppd.pok.ibm.com
fs2          sdbnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdcnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sddnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdensd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdgnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdfnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
fs2          sdhnsd      c5n94g.ppd.pok.ibm.com,c5n96g.ppd.pok.ibm.com
(free disk)  hd2n97      c5n97g.ppd.pok.ibm.com,c5n98g.ppd.pok.ibm.com
```

To find out the local device names for the disks, use the **mmlsnsd** command with the **-m** option. For example, issuing **mmlsnsd -m** produces output similar to this:

```
Disk name   NSD volume ID     Device            Node name             Remarks
-------------------------------------------------------------------------------
hd2n97      0972846145C8E924  /dev/hdisk2     c5n97g.ppd.pok.ibm.com  server node
hd2n97      0972846145C8E924  /dev/hdisk2     c5n98g.ppd.pok.ibm.com  server node
hd3n97      0972846145C8E927  /dev/hdisk3     c5n97g.ppd.pok.ibm.com  server node
hd3n97      0972846145C8E927  /dev/hdisk3     c5n98g.ppd.pok.ibm.com  server node
hd4n97      0972846145C8E92A  /dev/hdisk4     c5n97g.ppd.pok.ibm.com  server node
hd4n97      0972846145C8E92A  /dev/hdisk4     c5n98g.ppd.pok.ibm.com  server node
hd5n98      0972846245EB501C  /dev/hdisk5     c5n97g.ppd.pok.ibm.com  server node
hd5n98      0972846245EB501C  /dev/hdisk5     c5n98g.ppd.pok.ibm.com  server node
hd6n98      0972846245DB3AD8  /dev/hdisk6     c5n97g.ppd.pok.ibm.com  server node
hd6n98      0972846245DB3AD8  /dev/hdisk6     c5n98g.ppd.pok.ibm.com  server node
```

## Adding disks to a file system

Many file systems grow rapidly, so after creating a file system you might decide that more disk space is required.

Storage in a file system is divided in storage pools. The maximum size of any one disk that can be added to an existing storage pool is set approximately to the sum of the disk sizes when the storage pool is created. The actual value is shown in the **mmdf** command output.

Once a storage pool is created, the maximum size *cannot* be altered. However, you can create a new pool with larger disks, and then move data from the old pool to the new one.

When establishing a storage pool and when adding disks later to an existing storage pool, you should try to keep the sizes of the disks fairly uniform. GPFS allocates blocks round robin, and as the utilization level rises on all disks, the small ones will fill up first and all files created after that will be spread across fewer disks, which reduces the amount of prefetch that can be done for those files.

To add disks to a GPFS file system, first decide if you will:

1. Create new disks using the **mmcrnsd** command.

   In this case, you must also decide whether to create a new set of NSD and pools stanzas or use the rewritten NSD and pool stanzas that the **mmcrnsd** command produces. In a rewritten file, the disk usage, failure group, and storage pool values are the same as the values that are specified in the **mmcrnsd** command.
2. Select disks no longer in use in any file system. Issue the **mmlsnsd -F** command to display the available disks.

The disk may then be added to the file system using the stanza file as input to the **mmadddisk** command.

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

For more information, see the *mmadddisk command* and the *mmcrnsd command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting disks from a file system

Before deleting a disk use the **mmdf** command to determine whether there is enough free space on the remaining disks to store the file system.

**Note:** See "Querying file system space" on page 89 for more information about diagnosing space problems.

Consider how fragmentation may increase your storage requirements, especially when the file system contains a large number of small files. A margin of 150 percent of the size of the disks being deleted should be sufficient to allow for fragmentation when small files predominate. For example, in order to delete a 400 GB disk from your file system, which contains user home directories with small files, you should first determine that the other disks in the file system contain a total of 600 GB of free space.

If you do not replicate your file system data, you should rebalance the file system using the **mmrestripefs -b** command. If you replicate your file system data, run the **mmrestripefs -r** command after the disk has been deleted. This ensures that all data will still exist with correct replication after the disk is deleted. The **mmdeldisk** command only migrates data that would otherwise be lost, not data that will be left in a single copy.

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

Do not delete stopped disks, if at all possible. Start any stopped disk before attempting to delete it from the file system. If the disk cannot be started you will have to consider it permanently damaged. You will need to delete the disk using the appropriate **mmdeldisk** options. If metadata was stored on the disk, you will need to execute **mmfsck** in offline mode afterwards.. For more information on handling this situation, see *NSD and underlying disk subsystem failures* in the *IBM Spectrum Scale: Problem Determination Guide*.

When deleting disks from a file system, the disks may or may not be available. If the disks being deleted are still available, GPFS moves all of the data from those disks to the disks remaining in the file system. However, if the disks being deleted are damaged, either partially or permanently, it is not possible to move all of the data and you will receive I/O errors during the deletion process. For instructions on how to handle damaged disks, see *Disk media failure* in the *IBM Spectrum Scale: Problem Determination Guide*.

Specify the file system and the names of one or more disks to delete with the **mmdeldisk** command. For example, to delete the disk **hd2n97** from the file system **fs2** enter:

```
mmdeldisk fs2 hd2n97
```

The system displays information similar to:

```
Deleting disks ...
Scanning system storage pool
Scanning file system metadata, phase 1 ...
19 % complete on Fri Mar 16 23:23:50 2012
100 % complete on Fri Mar 16 23:23:51 2012
Scan completed successfully.
Scanning file system metadata, phase 2 ...
46 % complete on Fri Mar 16 23:23:55 2012
93 % complete on Fri Mar 16 23:23:58 2012
100 % complete on Fri Mar 16 23:23:58 2012
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
19.50 % complete on Fri Mar 16 23:24:25 2012 ( 35777 inodes 1207 MB)
47.92 % complete on Fri Mar 16 23:24:49 2012 ( 199955 inodes 2966 MB)
50.05 % complete on Fri Mar 16 23:25:09 2012 ( 235356 inodes 3098 MB)
53.09 % complete on Fri Mar 16 23:25:31 2012 ( 261831 inodes 3286 MB)
55.12 % complete on Fri Mar 16 23:25:51 2012 ( 283815 inodes 3412 MB)
63.25 % complete on Fri Mar 16 23:26:12 2012 ( 319236 inodes 3915 MB)
```

```
63.27 % complete on Fri Mar 16 23:26:33 2012 ( 382031 inodes 6223 MB)
63.29 % complete on Fri Mar 16 23:27:03 2012 ( 699858 inodes 9739 MB)
100.00 % complete on Fri Mar 16 23:27:35 2012
Scan completed successfully.
Checking Allocation Map for storage pool 'system'
17 % complete on Fri Mar 16 23:27:42 2012
31 % complete on Fri Mar 16 23:27:47 2012
48 % complete on Fri Mar 16 23:27:52 2012
62 % complete on Fri Mar 16 23:27:57 2012
76 % complete on Fri Mar 16 23:28:02 2012
90 % complete on Fri Mar 16 23:28:07 2012
100 % complete on Fri Mar 16 23:28:08 2012
tsdeldisk completed.
mmdeldisk: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

For syntax and usage information, refer to *mmdeldisk command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Replacing disks in a GPFS file system

Replacing an existing disk in a GPFS file system with a new one is the same as performing a delete disk operation followed by an add disk. However, this operation eliminates the need to restripe the file system following the separate delete disk and add disk operations as data is automatically moved to the new disk.

When replacing disks in a GPFS file system, first decide if you will:

1. Create new disks using the **mmcrnsd** command.

   In this case, you must also decide whether to create a new set of NSD and pools stanzas or use the rewritten NSD and pool stanzas that the **mmcrnsd** command produces. In a rewritten file, the disk usage, failure group, and storage pool values are the same as the values that are specified in the **mmcrnsd** command.
2. Select NSDs no longer in use by another GPFS file system. Issue the **mmlsnsd -F** command to display the available disks.

To replace a disk in the file system, use the **mmrpldisk** command. For example, to replace the NSD **hd3n97** in file system **fs2** with the existing NSD **hd2n97**, which is no longer in use by another file system, enter:

```
mmrpldisk fs2 hd3n97 hd2n97
```

The system displays information similar to:

```
Replacing hd3n97 ...

The following disks of fs2 will be formatted on node c33f2in01:
hd2n97: size 571398144 KB
Extending Allocation Map
Checking Allocation Map for storage pool 'system'
9 % complete on Fri Mar 16 23:33:29 2012
23 % complete on Fri Mar 16 23:33:34 2012
37 % complete on Fri Mar 16 23:33:40 2012
52 % complete on Fri Mar 16 23:33:45 2012
66 % complete on Fri Mar 16 23:33:50 2012
83 % complete on Fri Mar 16 23:33:55 2012
98 % complete on Fri Mar 16 23:34:00 2012
100 % complete on Fri Mar 16 23:34:00 2012
Completed adding disks to file system fs2.
Scanning system storage pool
Scanning file system metadata, phase 1 ...
13 % complete on Fri Mar 16 23:34:19 2012
100 % complete on Fri Mar 16 23:34:22 2012
Scan completed successfully.
```

```
Scanning file system metadata, phase 2 ...
29 % complete on Fri Mar 16 23:34:26 2012
67 % complete on Fri Mar 16 23:34:29 2012
100 % complete on Fri Mar 16 23:34:32 2012
Scan completed successfully.
Scanning file system metadata, phase 3 ...
Scan completed successfully.
Scanning file system metadata, phase 4 ...
Scan completed successfully.
Scanning user file metadata ...
8.21 % complete on Fri Mar 16 23:34:54 2012 ( 37741 inodes 770 MB)
22.65 % complete on Fri Mar 16 23:35:14 2012 ( 40182 inodes 2124 MB)
32.95 % complete on Fri Mar 16 23:35:34 2012 ( 160837 inodes 3090 MB)
35.15 % complete on Fri Mar 16 23:35:57 2012 ( 227991 inodes 3296 MB)
36.34 % complete on Fri Mar 16 23:36:17 2012 ( 265748 inodes 3408 MB)
37.34 % complete on Fri Mar 16 23:36:38 2012 ( 284398 inodes 3502 MB)
46.07 % complete on Fri Mar 16 23:37:04 2012 ( 310636 inodes 4320 MB)
61.41 % complete on Fri Mar 16 23:37:25 2012 ( 315141 inodes 5759 MB)
87.04 % complete on Fri Mar 16 23:37:50 2012 ( 350241 inodes 8163 MB)
87.06 % complete on Fri Mar 16 23:38:11 2012 ( 370562 inodes 10136 MB)
87.08 % complete on Fri Mar 16 23:38:42 2012 ( 392561 inodes 11982 MB)
87.10 % complete on Fri Mar 16 23:39:22 2012 ( 401049 inodes 13195 MB)
87.12 % complete on Fri Mar 16 23:40:14 2012 ( 1100590 inodes 15685 MB)
100.00 % complete on Fri Mar 16 23:40:57 2012
Scan completed successfully.
Checking Allocation Map for storage pool 'system'
10 % complete on Fri Mar 16 23:41:02 2012
26 % complete on Fri Mar 16 23:41:07 2012
33 % complete on Fri Mar 16 23:41:12 2012
44 % complete on Fri Mar 16 23:41:17 2012
68 % complete on Fri Mar 16 23:41:22 2012
100 % complete on Fri Mar 16 23:41:25 2012
Done
mmrpldisk: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

**Note:** If you attempt to replace a stopped disk and the file system is not replicated, the attempt will fail.

However, you can replace a stopped disk if the file system is replicated. You can do so in one of the following ways:

- Deletion, addition, and rebalancing method:
  1. Use the **mmdeldisk** command to delete the stopped disk from the file system.
  2. Use the **mmadddisk** command to add a replacement disk.
  3. Use the **mmrestripefs -b** command to rebalance the file system.

  While this method requires rebalancing, it returns the system to a protected state faster (because it can use all of the remaining disks to create new replicas), thereby reducing the possibility of losing data.

  —Or—
- Direct replacement method:

  Use the **mmrpldisk** command to directly replace the stopped disk.

  The **mmrpldisk** command only runs at single disk speed because all data being moved must be written to the replacement disk. The data is vulnerable while the command is running, and should a second failure occur before the command completes, it is likely that some data will be lost.

For more information on handling this situation, see *Disk media failure* in the *IBM Spectrum Scale: Problem Determination Guide*.

## Additional considerations for managing disks

If you delete, replace, or suspend a disk with strict replication enforced, you may receive an ENOSPC error when you create or append data to an existing file.

If you need to delete, replace, or suspend a disk and you need to write new data while the disk is offline, you can disable strict replication before you perform the disk action. However, data written while replication is disabled will not be properly replicated. Therefore, after you perform the disk action, you must re-enable strict replication and run the **mmrestripefs -r** command. To determine if a file system has strict replication enforced, issue the **mmlsfs -K** command.

## Displaying GPFS disk states

You can display the current state of one or more disks in your file system by issuing the **mmlsdisk** command.

The information includes parameters that were specified on the **mmcrfs** command, and the current availability and status of the disks. For example, to display the current status of the disk **hd8vsdn100** in the file system **fs1**, enter:

```
mmlsdisk fs1 -d hd8vsdn100
```

Status is displayed in a format similar to:

```
disk         driver   sector failure holds    holds                            storage
name         type       size   group metadata data  status        availability pool
------------ -------- ------ ------- -------- ----- ------------- ------------ ------------
hd8vsdn100   nsd         512       1 no       yes   ready         up           sp1
```

For syntax and usage information, see the *mmlsdisk command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Disk availability

The following information lists the possible values of disk availability, and what they mean.

A disk's availability determines whether GPFS is able to read and write to the disk. There are four possible values for availability:

**up** The disk is available to GPFS for normal **read** and **write** operations.

**down** No **read** and **write** operations can be performed on the disk.

**recovering**
An intermediate state for disks coming up, during which GPFS verifies and corrects data. **write** operations can be performed while a disk is in this state, but **read** operations cannot (because data on the disk being recovered might be stale until the **mmchdisk start** command completes).

**unrecovered**
The disk was not successfully brought up.

Disk availability is automatically changed from **up** to **down** when GPFS detects repeated I/O errors. You can also change the availability of a disk by issuing the **mmchdisk** command.

## Disk status

The following information lists the possible values for disk status, and what they mean.

Disk status controls data placement and migration. Status changes as a result of a pending delete operation, or when the **mmchdisk** command is issued to allow file rebalancing or re-replicating prior to disk replacement or deletion.

Disk status has seven possible values, but four are transitional:

**ready** Normal status.

**suspended**
**or**
**to be emptied**
Indicates that data is to be migrated off this disk.

**being emptied**
Transitional status in effect while a disk deletion is pending.

**emptied**
Indicates that data is already migrated off this disk.

**replacing**
Transitional status in effect for old disk while replacement is pending.

**replacement**
Transitional status in effect for new disk while replacement is pending.

GPFS allocates space only on disks with a status of **ready** or **replacement**.

GPFS migrates data off disks with a status of **being emptied**, **replacing**, **to be emptied**, or **suspended** onto disks with a status of **ready** or **replacement**. During disk deletion or replacement, data is automatically migrated as part of the operation. Issue the **mmrestripefs** command to initiate data migration from a suspended disk.

See "Deleting disks from a file system" on page 114, "Replacing disks in a GPFS file system" on page 116, and "Restriping a GPFS file system" on page 88.

## Changing GPFS disk states and parameters

You might find it necessary to change a disk's state if there is some indication of disk failure or if you need to restripe the file system.

Refer to "Displaying GPFS disk states" on page 118 for a detailed description of disk states. You can change both the availability and status of a disk using the **mmchdisk** command:

- Change disk availability using the **mmchdisk** command and the **stop** and **start** options
- Change disk status using the **mmchdisk** command and the **suspend** and **resume** options.

Issue the **mmchdisk** command with one of the following four options to change disk state:

**resume**
Informs GPFS that a disk previously suspended is now available for allocating new space. Resume a disk only when you've suspended it and decided not to delete or replace it. If the disk is currently in a stopped state, it remains stopped until you specify the **start** option. Otherwise, normal **read** and **write** access to the disk resumes.

**start**
Informs GPFS that a disk previously stopped is now accessible. GPFS does this by first changing the disk availability from **down** to **recovering**. The file system metadata is then scanned and any missing updates (replicated data that was changed while the disk was down) are repaired. If this operation is successful, the availability is then changed to **up**.

If the metadata scan fails, availability is set to **unrecovered**. This could occur if other disks remain in **recovering** or an I/O error has occurred. Repair all disks and paths to disks. It is recommended that you run **mmfsck** at this point (For more information, see *mmfsck command* in the *IBM Spectrum Scale: Command and Programming Reference*). The metadata scan can then be re-initiated at a later time by issuing the **mmchdisk start** command again.

If more than one disk in the file system is down, they should all be started at the same time by using the **-a** option. If you start them separately and metadata is stored on any disk that remains down, the **mmchdisk start** command fails.

**stop**
Instructs GPFS to stop any attempts to access the specified disk. Use this option to inform GPFS that a disk has failed or is currently inaccessible because of maintenance. A disk's availability remains **down** until it is explicitly started with the **start** option.

**suspend**
**or**
**empty**
Instructs GPFS to stop allocating space on the specified disk. Place a disk in this state prior to disk deletion or replacement. This is a user-initiated state that GPFS will never use without an explicit command to change disk state.

**Note:** A disk remains suspended until it is explicitly resumed. Restarting GPFS or rebooting nodes does not restore normal access to a suspended disk.

The `empty` option is similar to the `suspend` option. For releases prior to GPFS 4.1.1, the output of the **mmlsdisk** command displays the status as `suspended`, as shown in the following example.

For example, to suspend the **hd8vsdn100** disk in the file system **fs1**, enter:

```
mmchdisk fs1 suspend -d hd8vsdn100
```

To confirm the change, enter:

```
mmlsdisk fs1 -d hd8vsdn100
```

The system displays information similar to:

```
disk         driver   sector failure holds    holds                            storage
name         type       size   group metadata data  status        availability pool
------------ -------- ------ ------- -------- ----- ------------- ------------ ------------
hd8vsdn100   nsd         512       7 yes      yes   suspended     up           system
```

From GPFS 4.1.1 onwards, the status in the **mmlsdisk** command is displayed as `to be emptied`, as shown in the following example:

For example, to empty the **gpfs1nsd** disk in the file system **fs1**, enter:

```
mmchdisk fs1 empty -d gpfs1nsd
```

To confirm the change, enter:

```
mmlsdisk fs1 -d gpfs1nsd
```

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
--------- ------ ------ ------ --------  ----  ------------- ------------ ------- -------- ---------
gpfs1nsd  nsd     512      -1   Yes       Yes   to be emptied   up             1   system
gpfs2nsd  nsd     512      -1   Yes       Yes   to be emptied   up             2   system    desc
```

You can also use the **mmchdisk** command with the **change** option to change the *Disk Usage* and *Failure Group* parameters for one or more disks in a GPFS file system. This can be useful in situations where, for example, a file system that contains only RAID disks is being upgraded to add conventional disks that are better suited to storing metadata. After adding the disks using the **mmadddisk** command, the metadata currently stored on the RAID disks would have to be moved to the new disks to achieve the desired performance improvement. To accomplish this, first the **mmchdisk change** command would be issued to change the *Disk Usage* parameter for the RAID disks to **dataOnly**. Then the **mmrestripefs** command would be used to restripe the metadata off the RAID device and onto the conventional disks.

For example, to specify that metadata should no longer be stored on disk **hd8vsdn100**, enter:

```
mmchdisk fs1 change -d "hd8vsdn100:::dataOnly"
```

To confirm the change, enter:

```
mmlsdisk fs1 -d hd8vsdn100
```

The system displays information similar to:

```
disk         driver   sector failure holds    holds                            storage
name         type       size   group metadata data  status        availability pool
------------ -------- ------ ------- -------- ----- ------------- ------------ ------------
hd8vsdn100   nsd         512       1 no       yes   ready         up           sp1
```

For complete usage information, see the *mmchdisk command* and the *mmlsdisk command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Changing your NSD configuration

Use the following steps to change the NSD configuration.

Once your NSDs have been created, you may change the configuration attributes as your system requirements change. For more information about creating NSDs, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* and the *mmcrnsd command* in the *IBM Spectrum Scale: Command and Programming Reference*)

By issuing the **mmchnsd** command you can:

- Specify up to eight servers for an NSD that does not have one.
- Change the NSD server nodes specified in the server list.
- Delete the server list. The disk must now be SAN-attached to all nodes in the cluster on which the file system will be mounted.

You must follow these rules when changing NSDs:

- Identify the disks by the NSD names that were given to them by the **mmcrnsd** command.
- Explicitly specify values for all NSD servers in the list, even if you are only changing one of the values.
- Unmount the file system that contains the NSD being changed prior to issuing the **mmchnsd** command.
- Connect the NSD to the new nodes prior to issuing the **mmchnsd** command.
- **mmchnsd** cannot change the *DiskUsage* or *FailureGroup* for an NSD. Use the **mmchdisk** command to change these attributes.
- To move a disk from one storage pool to another, use the **mmdeldisk** and **mmadddisk** commands.
- You cannot change the name of the NSD.

For example, to assign node **k145n07** as an NSD server for disk **gpfs47nsd**:

1. Make sure that **k145n07** is not already assigned to the server list by issuing the **mmlsnsd** command.

   ```
   mmlsnsd -d "gpfs47nsd"
   ```

   The system displays information similar to:

   ```
   File system   Disk name    NSD server nodes
   ------------------------------------------------------------
   fs1           gpfs47nsd    k145n09
   ```

2. Unmount the file system on all nodes and ensure that the disk is connected to the new node (**k145n07**).
3. Issue the **mmchnsd** command:

```
mmchnsd "gpfs47nsd:k145n09,k145n07"
```

4. Verify the changes by issuing the **mmlsnsd** command:

```
mmlsnsd -d gpfs47nsd
```

The system displays information similar to:

```
  File system   Disk name    NSD servers
---------------------------------------------------------------------------
 fs2           gpfs47nsd    k145n09.ppd.pok.ibm.com,k145n07.ppd.pok.ibm.com
```

## Changing NSD server usage and failback

GPFS determines if a node has physical or virtual connectivity to an underlying NSD disk through a sequence of commands invoked from the GPFS daemon. This determination is called disk discovery and occurs at both initial GPFS startup as well as whenever a file system is mounted.

The default order of access used in disk discovery:

1. Local block device interfaces for SAN, SCSI or IDE disks
2. NSD servers

The **useNSDserver** file system mount option can be used to set the order of access used in disk discovery, and limit or eliminate switching from local access to NSD server access, or the other way around. This option is specified using the **-o** flag of the **mmmount**, **mount**, **mmchfs**, and **mmremotefs** commands, and has one of these values:

**always**
: Always access the disk using the NSD server.

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

**asneeded**
: Access the disk any way possible. This is the default.

**never**
: Always use local disk access.

For example, to always use the NSD server when mounting file system **fs1**, issue this command:

```
mmmount fs1 -o useNSDserver=always
```

To change the disk discovery of a file system that is already mounted: cleanly unmount it, wait for the unmount to complete, and then mount the file system using the desired **-o useNSDserver** option.

## Enabling and disabling Persistent Reserve

GPFS can use Persistent Reserve (PR) functionality to improve failover times (with some restrictions).

The following restrictions apply to the use of PR:

- PR is supported on both AIX and Linux nodes. However, note the following:
  - If the disks have defined NSD servers, then the NSD server nodes must all be running AIX, or they must all be running Linux.
  - If the disks are SAN-attached to all nodes, then the SAN-attached nodes in the cluster must all be running AIX, or they must all be running Linux.
- The disk subsystems must support PR
- GPFS supports a mix of PR disks and other disks. However, you will only realize improved failover times if **all** disks in the cluster support PR.

- GPFS only supports PR in the local cluster. Remote mounts must access the disks through an NSD server.
- When you enable or disable PR, you must stop GPFS on all nodes.
- Before enabling PR, make sure all disks are in the same initial state.

To enable (or disable) Persistent Reserve, issue the command:

```
mmchconfig usePersistentReserve={yes|no}
```

For fast recovery times with Persistent Reserve, you should also set the *failureDetectionTime* configuration parameter. For fast recovery, a recommended value would be 10. You can set this by issuing the command:

```
mmchconfig failureDetectionTime=10
```

To determine if the disks on the servers and the disks of a specific node have PR enabled, issue the following command from the node:

```
mmlsnsd -X
```

The system responds with something similar to:

```
Disk name  NSD volume ID     Device      Devtype  Node name          Remarks
---------------------------------------------------------------------------------------
gpfs10nsd  09725E5E43035A99  /dev/hdisk6  hdisk  k155n14.kgn.ibm.com  server node,pr=yes
gpfs10nsd  09725E5E43035A99  /dev/hdisk8  hdisk  k155n16.kgn.ibm.com  server node,pr=yes
gpfs10nsd  09725E5E43035A99  /dev/hdisk6  hdisk  k155n17.kgn.ibm.com  directly attached pr=yes
```

If the GPFS daemon has been started on all the nodes in the cluster and the file system has been mounted on all nodes that have direct access to the disks, then **pr=yes** should be on all hdisks. If you do not see this, there is a problem. Refer to the *IBM Spectrum Scale: Problem Determination Guide* for additional information on Persistent Reserve errors.

# Chapter 13. Managing protocol services

GPFS provides system administrators with the ability to manage the protocol services.

## Configuring and enabling SMB and NFS protocol services

If you have not previously enabled and started the Cluster Export Services (CES) protocol services, enable and start them now.

### Prerequisites

When you enable SMB protocol services, the following prerequisites must be met:

- The number of CES nodes must be 16 or lower.
- All CES nodes must be running the same system architecture. For example, mixing nodes based on Intel and Power is not supported.

When you add new CES nodes to a running system where the SMB protocol is enabled, the following prerequisite must be met:

- All CES nodes must be in `SMB HEALTHY` state.

When you remove a CES node from a running system where the SMB protocol is enabled, the following prerequisite must be met:

- All CES nodes (except for the node that is being removed) must be in `SMB HEALTHY` state.

For more information about the SMB states, see **`mmces command`** in *IBM Spectrum Scale: Command and Programming Reference*.

Disabling SMB protocol services deletes all configured SMB shares and SMB settings. After re-enabling SMB, all exports and settings must be re-created and reconfigured.

### Enabling protocol services

Issue the following commands to enable SMB and NFS services on all CES nodes:

-
```
mmces service enable SMB
```
-
```
mmces service enable NFS
```

**GUI navigation**

- To enable SMB services in the GUI, log on to the IBM Spectrum Scale GUI and select **Settings** > **SMB Service**.
- To enable NFS services in the GUI, log on to the IBM Spectrum Scale GUI and select **Settings** > **NFS Service**.

The protocol services that are used need to be started on all CES nodes:

```
mmces service start SMB -a
mmces service start NFS -a
```

After you start the protocol services, verify that they are running by issuing the **`mmces state show`** command.

**Note:** The start and stop are maintenance commands. Stopping a service on a particular protocol node, without first suspending the node ensures the public IP addresses on that node stay with that node. In this event, protocol clients might attempt to connect using these IP addresses and fail to connect to the service.

The sequence of commands to enable file access and then disable file access, using the NFS service as an example, follows:

1. Enable NFS by using the following command:

   ```
   mmces service enable NFS
   ```

   **Note:** This command also starts NFS on all CES nodes.

   Then, you need to set up the authentication before you can add an export. The easiest authentication setup is to use system authentication.
2. Set up the authentication by using `userdefined` authentication type and `file` data access method:

   ```
   mmuserauth service create --data-access-method file --type userdefined
   ```
3. Add an export by running the following command, where **fs0** is a GPFS file system and **fset0** is an independent fileset:

   ```
   mmnfs export add /gpfs/fs0/fset0
   ```
4. Verify that this is configured and running by using the following commands:

   ```
   mmces service list -a
   mmuserauth service list
   mmnfs export list
   ```
5. Stop NFS and disable NFS protocol on the CES nodes by running the following commands:

   ```
   mmces service stop nfs -a
   mmuserauth service remove --data-access-method file
   ```

   **Note:** Authentication services must be removed for file access before disabling the NFS; and if SMB is enabled and running, then you cannot remove file authentication now.
6. Disable NFS service on the CES nodes by running the following command:

   ```
   mmces service disable NFS
   ```

   **Note:** The NFS configuration is removed when NFS is disabled, so whatever was exported previously is lost. If you want to save the NFS configuration, you should backup the contents of `/var/mmfs/ces/nfs-config/` on any protocol node.

## Configuring and enabling the Object protocol service

If you want to use the Cluster Export Services (CES) object service and it was not configured and enabled during the installation, configure object services now.

1. If a file system for the object data has not been created yet, you must create it now (see **mmcrfs command** in *IBM Spectrum Scale: Command and Programming Reference*).
2. Use the **mmobj** command for the initial configuration of the object stack.

   **Note:** A separate fileset will be created to hold the object data:

   ```
   mmobj swift base -g /gpfs/fs01 --cluster_hostname clustername --local_keystone --db_password Passw0rd
   --admin_password Password -i 2000 --enable-s3
   ```

   This example creates a fileset for Object storage in the `/gpfs/fs01` fileset with the specified hostname as access point and 2000 inodes. A local database is created for Keystone authentication and S3 API is enabled. See **mmobj command** in *IBM Spectrum Scale: Command and Programming Reference* for details.
3. After the initial configuration, enable and start the object services by running the following commands:

   ```
   mmces service enable OBJ
   ```

4. Verify that the object service is running as expected:
   ```
   mmces service list -a
   ```

## Performance tuning for object services

By default, the IBM Spectrum Scale installation sets the number of workers for the object services low. These numbers can be adjusted upwards if you have protocol servers with sufficient cores and memory.

As with most performance tuning, there is no single correct setting. A good starting point for tuning worker counts is to set workers in the `proxy-server.conf` to `auto` so that one worker is started for every core on a protocol node. The other servers can be set to a percentage of the number of cores on your protocol nodes:

- object server set to 75% of core count
- container server set to 50% of core count
- account server set to 25% of core count

Depending on the load of other protocol workloads, the optimal settings for worker count might be higher or lower than this on your system.

For example, if you have 16 cores in your protocol nodes, the following commands can be used to tune your worker settings:

```
mmobj config change --ccrfile proxy-server.conf --section DEFAULT --property workers --value auto

mmobj config change --ccrfile object-server.conf --section DEFAULT --property workers --value 12

mmobj config change --ccrfile container-server.conf --section DEFAULT --property workers --value 8

mmobj config change --ccrfile account-server.conf --section DEFAULT --property workers --value 4
```

---

## Disabling protocol services

If a protocol service is no longer needed, it can be disabled by using the **mmces** command.

To disable a protocol service, enter the appropriate **mmces** command:

- `mmces service disable SMB`

  **Note:** SMB can be removed only when authentication has been cleared (or is user defined).

  Disabling SMB services will stop SMB on all CES nodes and remove the SMB configuration from the Cluster Configuration Repository (CCR), the SMB clustered databases (tdb's), and the SMB-related config files in **/var/mmfs/ces** on the CES nodes.
- `mmces service disable NFS`

  **Note:** Disabling NFS services will stop NFS on all CES nodes and remove the NFS configuration from the Cluster Configuration Repository (CCR) and remove /var/mmfs/ces/nfs-config/ on the CES nodes.
- `mmces service disable OBJ`

  **Note:** For information on disabling the Object services, see "Understanding and managing Object services" on page 183.

# Chapter 14. Managing protocol user authentication

The system administrator can configure authentication for both object and file access either during the installation of the system or after the installation. If the authentication configuration is not configured during installation, you can manually do it by using the **mmuserauth service create** command from any node in the IBM Spectrum Scale cluster. This section covers the manual method of configuring authentication for file and object access.

## Setting up authentication servers to configure protocol user access

Before you start configuring authentication for protocol access, the system administrator needs to ensure that the authentication server is set up properly and the connection between the IBM Spectrum Scale system and authentication server is established properly.

Depending on the requirement, the IBM Spectrum Scale system administrator needs to set up the following servers:

- Microsoft Active Directory (AD) for file and object access
- Lightweight Directory Access Protocol server for file and object access
- Keystone server to configure local, AD, or LDAP-based authentication for object access. Configuring Keystone is a mandatory requirement if you need to have Object access.

AD and LDAP servers are set up externally. You can configure either an internal or external Keystone server. The installation and configuration of an external authentication server must be handled separately. The IBM Spectrum Scale system installation manages the installation and set up of internal Keystone server.

### Integrating with AD server

If the authentication method is selected as AD, the customer must set up the AD server before configuring the authentication method in the IBM Spectrum Scale system.

Ensure that you have the following details before you start configuring AD-based authentication:

- IP address or host name of the AD server.
- DNS is configured on all protocol nodes of the system, where the primary DNS needs to be configured as the AD domain controller.
- Domain details such as the following:
  - Domain name and realm.
  - AD admin user ID and password to join the IBM Spectrum Scale system as machine account into the AD domain.
- ID map role of the system is identified.
- Define the ID map range and size depending upon the maximum RID (sum of allocated and expected growth).
- Primary DNS is added in the /etc/resolv.conf file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. This is a mandatory requirement when AD is used as the authentication server as the DNS must be able to resolve the host domain and its trusted domains of interest. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, see the corresponding Operating System documentation.

- During the AD join process, a computer account having the same name as the netbios name is searched within the AD domain that will be joined. If the name is not found, a new computer entry is created in the standard location (CN=Computers). If the user chooses to pre-create computer accounts for IBM Spectrum Scale in the AD domain within a particular organizational unit, the computer account must be created with a valid name and it must be passed as the netbios name while configuring the IBM Spectrum Scale system. After the account is created on the AD server, the system must be joined to the AD domain.

To achieve high-availability, you can configure multiple AD domain controllers. While configuring AD-based authentication, you do not need to specify multiple AD servers in the command line to achieve high-availability. The IBM Spectrum Scale system queries the specified AD server for relevant details and configures itself for the AD-based authentication. The IBM Spectrum Scale system relies on the DNS server to identify the set of available AD servers that are currently available in the environment serving the same domain system.

## Integrating with LDAP server

If LDAP-based authentication is selected, ensure that the LDAP server is set up with the required schemas to handle the authentication and ID mapping requests. If you need to support SMB data access, LDAP schema must be extended before configuring the authentication.

Ensure that you have the following details before you start configuring LDAP based authentication:

- Domain details such as base dn, and dn prefixes of groups and users, else default values are used. Default user group suffix is `<ou=Groups,<base dn>` and default user suffix is `ou=People,<base dn>`.
- IP address or host name of LDAP server.
- Admin user ID and password of LDAP server that is used during LDAP simple bind and for LDAP searches.
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.
- NetBIOS name that is to be assigned for the IBM Spectrum Scale system.
- If you need to have secure communication between the IBM Spectrum Scale system and LDAP, the CA signed certificate that is used by the LDAP server for TLS communication must be placed at the specified location in the system.
- If you are using LDAP with Kerberos, create Kerberos keytab file by using the MIT KDC infrastructure.
- Primary DNS is added in the /etc/resolv.conf file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, refer the corresponding Operating System documentation.

### Setting up LDAP server prerequisites

Before you start configuring the IBM Spectrum Scale system with LDAP server, the following external LDAP server prerequisites must be met:

- The LDAP server must already be configured.
- Enable TLS encryption on the LDAP server, if you need to secure communication between the IBM Spectrum Scale system and LDAP server. Details on configuring SSL or TLS encryption on the server can be obtained from the *OpenLDAP Administrator's Guide*.
- To access SMB shares, LDAP user information must be updated with unique Samba attributes in addition to the attributes that are stored for a normal LDAP user. Ensure that these required Samba attributes are present in the LDAP user entries.
- Ensure you do not have the same user name for different organizational units of the LDAP server that is configured with the IBM Spectrum Scale system.

**LDAP bind user requirements:**

When an IBM Spectrum Scale system is configured with LDAP as the authentication method, the IBM Spectrum Scale system needs to connect to the LDAP server by using an administrative user ID and password. This administrative user is referred as bind user.

It is recommended that the bind user is given enough privileges that are required by the storage system to mitigate any security concerns.

This bind user must at least have permission to query users and groups that are defined in the LDAP server to allow storage system to authenticate these users. The bind user information (`bind dn`) is also used by Samba server while making LDAP queries to retrieve required information from the LDAP server.

**Note:** In the following sections, it is assumed that the user account for the bind user exists in the LDAP directory server. The bind user distinguished name (also known as dn) used in the following examples is `uid=ibmbinduser,ou=people,dc=ldapserver,dc=com`. This name needs to be updated based on the bind user that is used with the IBM Spectrum Scale system.

*OpenLDAP server ACLs:*

The OpenLDAP server ACLs define the privileges that are required for the bind user.

The following example uses ACLs that are required for the bind user and other type of users for the sake of completeness. It is likely that a corporate directory server has those ACLs configured already and only the entries for bind user need to be merged correctly in the `slapd` configuration file (generally, `/etc/openldap/slapd.conf` file on Linux systems). Follow the ACL ordering rules to ensure that correct ACLs are applied.

```
### some attributes need to be readable so that commands like 'id user',
'getent' etc can answer correctly.
access to attrs=cn,objectClass,entry,homeDirectory,uid,uidNumber,
gidNumber,memberUid
by dn="uid=ibmbinduser,ou=people,dc=ldapserver,dc=com" read

###The following will not list userPassword when ldapsearch is
performed with bind user.

### Anonymous is needed to allow bind to succeed and users to
authenticate, should be
a pre-existing entry already.
access to attrs=userPassword
    by dn="uid=ibmbinduser,ou=people,dc=ldapserver,dc=com"  auth
    by self write
    by anonymous auth
    by * none

### Storage system needs to be able to find samba domain account
specified on the mmuserauth service create command.

###It is strongly recommended that domain account is pre-created
to ensure

###consistent access to multiple storage systems.

###Uncomment ONLY if you want storage systems to create domain
account when it does not exist.
#access to dn.base="dc=ldapserver,dc=com"
#    by dn="uid=ibmbinduser,ou=people,dc=ldapserver,dc=com " write
#    by * none
```

```
access to dn.regex="sambadomainname=[^,]+,dc=ldapserver,dc=com"
    by dn=" uid=ibmbinduser,ou=people,dc=ldapserver,dc=com" read
    by * none

### all samba attributes need to be readable for samba access
access to attrs=cn,sambaLMPassword,sambaNTPassword,sambaPwdLastSet,
sambaLogonTime,sambaLogoffTime,sambaKickoffTime,sambaPwdCanChange,
sambaPwdMustChange,sambaAcctFlags,displayName,sambaHomePath,
sambaHomeDrive,sambaLogonScript,sambaProfilePath,description,
sambaUserWorkstations,sambaPrimaryGroupSID,sambaDomainName,
sambaMungedDial,sambaBadPasswordCount,sambaBadPasswordTime,
sambaPasswordHistory,sambaLogonHours,sambaSID,sambaSIDList,
sambaTrustFlags,sambaGroupType,sambaNextRid,sambaNextGroupRid,
sambaNextUserRid,sambaAlgorithmicRidBase,sambaShareName,
sambaOptionName,sambaBoolOption,sambaIntegerOption,
sambaStringOption,sambaStringListoption
    by dn="uid=ibmbinduser,ou=people,dc=ldapserver,dc=com" read
    by self read
    by * none

### Need write access to record bad failed login attempt
access to attrs=cn,sambaBadPasswordCount,sambaBadPasswordTime,
sambaAcctFlags by dn="uid=ibmbinduser,ou=people,dc=ldapserver,
dc=com" write

### Required to check samba schema
access to dn.base=* by  dn="uid=ibmbinduser,ou=people,
dc=ldapserver,dc=com" read
```

*IBM Spectrum Protect Directory Server ACLs:*

The IBM Spectrum Protect Directory Server ACLs define the privileges that are required for the bind user, when using IBM Spectrum Protect Directory Server.

These ACLs are provided in the LDIF format and can be applied by submitting the **ldapmodify** command.

```
dn: dc=ldapserver,dc=com
changetype: modify

add: ibm-filterAclEntry

ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapserver,dc=com:
(objectClass=sambaSamAccount):normal:rsc:sensitive:rsc:critical:rsc
-
add:ibm-filterAclEntry

ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapserver,dc=com:
(objectclass=sambaDomain):normal:rwsc:sensitive:rwsc:critical:rwsc

 dn:uid=ibmbinduser,ou=people,dc=ldapserver,dc=com

add:aclEntry

aclentry: access-id:uid=ibmbinduser,ou=people,dc=ldapserver,dc=com:at.cn:r:at.
objectClass:r:at.homeDirectory:r:at.uid:r:at.uidNumber:s:

at.gidNumber:r:at.memberUid:r:at.userPassword:sc:at.sambaLMPassword:r:at.
sambaNTPassword:r:at.sambaPwdLastSet:r:at.sambaLogonTime:r:

at.sambaLogoffTime:r:at.sambaKickoffTime:r:at.sambaPwdCanChange:r:at.
sambaPwdMustChange:r:at.sambaAcctFlags:r:at.displayName:r:
```

```
at.sambaHomePath:r:at.sambaHomeDrive:r:at.sambaLogonScript:r:at.sambaProfilePath:
r:at.description:r:at.sambaUserWorkstations:r:

at.sambaPrimaryGroupSID:r:at.sambaDomainName:r:at.sambaMungedDial:r:at.
sambaBadPasswordCount:r:at.sambaBadPasswordTime:r:
at.sambaPasswordHistory:r:at.sambaLogonHours:r:at.sambaSID:r:at.sambaSIDList:r:at.
sambaTrustFlags:r:at.sambaGroupType:r:
at.sambaNextRid:r:at.sambaNextGroupRid:r:at.sambaNextUserRid:r:at.
sambaAlgorithmicRidBase:r:at.sambaShareName:r:at.sambaOptionName:r:

at.sambaBoolOption:r:at.sambaIntegerOption:r:at.sambaStringOption:r:at.
sambaStringListoption:r:at.sambaBadPasswordCount:rwsc:

at.sambaBadPasswordTime:rwsc:at.sambaAcctFlags:rwsc

### Storage system needs to be able to find samba domain account specified
on the mmuserauth service create command.

###It is strongly recommended that domain account is pre-created to ensure

###consistent access to multiple storage systems.

###Uncomment ONLY if you want storage systems to create domain account when
it does not exist.

 dn: dc=ldapserver,dc=com

changetype: modify

add:ibm-filterAclEntry

ibm-filterAclEntry:access-id:uid=ibmbinduser,ou=people,dc=ldapserver,
dc=com:(objectclass=domain):object:grant:a
```

See *IBM Tivoli Directory Server Administration Guide* for information about applying these ACLs on the IBM Spectrum Protect Directory Server.

## Updating LDAP user information with Samba attributes

If you need to support SMB data access, LDAP schema must be extended to store more attributes such as SID, Windows password hash to the POSIX user object. To use Samba accounts, update LDAP user information with unique Samba attributes.

The following sample LDIF file shows the minimum required samba attributes:

```
dn: cn=SMBuser,ou=People,dc=ibm,dc=com
changetype: modify
add : objectClass
objectClass: sambaSamAccount
-
add: sambaSID
sambaSID: S-1-5-21-1528920847-3529959213-2931869277-1102
-
add:sambaPasswordHistory
sambaPasswordHistory: 0000000000000000000000000000000000000000000000000000000000000000
-
add:sambaNTPassword
sambaNTPassword: (valid samba password hash )
-
add:sambaPwdLastSet
```

```
sambaPwdLastSet: 1263386096
-
add:SambaAcctFlags
sambaAcctFlags: [U          ]
```

**Note:** Attributes must be separated with a dash as the first and only character on a separate line.

Perform the following steps to create the values for `sambaNTPassword`, `sambaPwdLastSet`, and `SambaAcctFlags`, which must be generated from a PERL module:

1. Download the module from http://search.cpan.org/~bjkuit/Crypt-SmbHash-0.12/SmbHash.pm. Create and install the module by following the readme file.
2. Use the following PERL script to generate the LM and NT password hashes:

```
# cat /tmp/Crypt-SmbHash-0.12/gen_hash.pl
#!/usr/local/bin/perl
use Crypt::SmbHash;
$username = $ARGV[0];
$password = $ARGV[1];
if ( !$password ) {
        print "Not enough arguments\n";
        print "Usage: $0 username password\n";
        exit 1;
}
$uid = (getpwnam($username))[2];
my ($login,undef,$uid) = getpwnam($ARGV[0]);
ntlmgen $password, $lm, $nt;
printf "%s:%d:%s:%s:[%-11s]:LCT-%08X\n", $login, $uid, $lm, $nt, "U", time;
```

3. Generate the password hashes for any user as in the following example for the user test01:

```
# perl gen_hash.pl SMBuser test01

:0:47F9DBCCD37D6B40AAD3B435B51404EE:82E6D500C194BA5B9716495691FB7DD6:
[U          ]:LCT-4C18B9FC
```

   **Note:** The output contains login name, uid, LM hash, NT hash, flags, and time, with each field separated from the next by a colon. The login name and uid are omitted because the command was not run on the LDAP server.

4. Use the information from step 3 to update the LDIF file in the format that is provided in the example at the beginning of this topic.
   - To generate the `sambaPwdLastSet` value, use the hexadecimal time value from step 3 after the dash character and convert it into decimal.
   - A valid samba SID is required for a user to enable that user's access to an IBM Spectrum Scale share. To generate the samba SID, multiply the user's UID by 2 and add 1000. The users's SID must contain the samba SID from the `sambaDomainName`, which is either generated or picked up from the LDAP server, if it exists. The following attributes for `sambaDomainName` LDIF entry are required:

     ```
     dn: sambaDomainName=(IBM Spectrum Scale system),dc=ibm,dc=com
     sambaDomainName: (IBM Spectrum Scale system name)
     sambaSID: S-1-5-21-1528920847-3529959213-2931869277
     sambaPwdHistoryLength: 0
     sambaMaxPwdAge: -1
     sambaMinPwdAge: 0
     ```

     This entry can be created by the LDAP server administrator by using either of the following two methods:

     – Write and run a bash script similar to the following example:

     ```
     sambaSID=
             for num in 1 2 3 ;do
                 randNum=$(od -vAn -N4 -tu4 < /dev/urandom | sed -e 's/ //g')
                 if [ -z     "$sambaSID" ];then
                     sambaSID="S-1-5-21-$randNum"
                 else
     ```

```
                sambaSID="${sambaSID}-$    {randNum}"
            fi
        done
echo $sambaSID
```

Then, use the samba SID generated to create the LDIF file. The `sambaDomainName` must match the IBM Spectrum Scale system name.

– When you run the **mmuserauth service create** command, the system creates the `sambaDomainName`, if it does not exist.

The `sambaSID` for every user must have the following format: `(samba SID for the domain)-(userID*2+1000)`. For example: `S-1-5-21-1528920847-3529959213-2931869277-1102`

**Note:** To enable access to more than one IBM Spectrum Scale system, the domain SID prefix of all of the systems must match. If you change the domain SID for an IBM Spectrum Scale system on the LDAP server, you must restart CTDB on that IBM Spectrum Scale system for the change to take effect.

5. Submit the **ldapmodify** command as shown in the following example to update the user's information :

```
# ldapmodify -h localhost -D cn=Manager,dc=ibm,dc=com -W -x -f /tmp/samba_user.ldif
```

## Integrating with Keystone Identity Service

The object protocol uses the keystone service to authenticate object users. When configuring the IBM Spectrum Scale system, you must specify that either an internal keystone server or an external keystone server will be used. In either case, the keystone server can use a local database or a separate LDAP or AD system for managing user credentials. If you are using an external keystone server, you are responsible for the configuration of this service. For more information, refer to the OpenStack documentation.

Before you configure authentication for object, ensure that the object services are enabled. To enable object services, use the **mmces service enable obj** command.

### Prerequisites

Ensure that you have the following details before you start configuring local authentication for object access:

- The keystone host name must be defined and configured on all protocol nodes of the cluster. This host name returns one of the CES IP addresses, such as a round-robin DNS. It could also be a fixed IP of a load balancer that distributes requests to one of the CES nodes. This host name is also used to create the keystone endpoints.
- If you want the keystone server to use external CA signed certificate for signing the keystone generated tokens, ensure that the following certificates are available in `/var/mmfs/tmp` directory of the node where you run the commands:
  – /var/mmfs/tmp/signing_cert.pem
  – /var/mmfs/tmp/signing_key.pem
  – /var/mmfs/tmp/signing_cacert.pem

  **Note:** By default, the system uses an internal self-signed certificate that is generated by **keystone-manage pki_setup**. For production environments we recommend the use of certificates issued by a trusted CA rather than self-signed certificates.
- If you want to enable SSL on keystone, ensure that the following certificates that are placed on the `/var/mmfs/tmp` directory of the node where commands are run:
  – /var/mmfs/tmp/ssl_cert.pem
  – /var/mmfs/tmp/ssl_key.pem
  – /var/mmfs/tmp/ssl_cacert.pem
  – /var/mmfs/tmp/ks_ext_cacert.pem

**Note:** By default, the IBM Spectrum Scale installation process configures object authentication with a local keystone authentication method.

## Configuring authentication and ID mapping for file access

The system administrator can decide whether to configure authentication and ID mapping method either during the installation of the IBM Spectrum Scale system or after the installation. If the authentication configuration is not configured during installation, you can manually do it by using the **mmuserauth service create** command from any protocol node of the IBM Spectrum Scale system. This section covers the manual method of configuring authentication for file access.

You can configure the following external authentication servers for file access:

- Active Directory (AD)
- Light Weight Directory Access Protocol (LDAP)
- Network Information Service (NIS)

Ensure that the following RPMs are installed on all protocol nodes before you start configuring the authentication method:

**On Red Hat Enterprise Linux nodes**

- **For AD:**
  - `bind-utils`

  **Note:** If you are planning to use AD for file authentication, you must stop the nscd service by using the **service nscd stop** command before installation.
- **For LDAP:**
  - `openldap-clients`
  - `sssd` and its dependencies ( particularly `sssd-common` and `sssd-ldap`)
- **For NIS:**
  - `sssd` and its dependencies ( particularly `sssd-common` and `sssd-proxy`)
  - `ypbind` and its dependencies (`yp-tools`)

**On SLES nodes**

- **For AD:**
  - `bind-utils`

  **Note:** If you are planning to use AD for file authentication, you must stop the nscd service by using the **service nscd stop** command before installation.
- **For LDAP:**
  - `openldap2-clients`
  - `sssd` and its dependencies ( particularly `sssd-common` and `sssd-ldap`)
- **For NIS:**
  - `sssd` and its dependencies ( particularly `sssd-common` and `sssd-proxy`)
  - `ypbind` and its dependencies (`yp-tools`)

## AD-based authentication for file access

You can configure Microsoft Active Directory (AD) as the authentication server to manage the authentication requests and to store user credentials.

You can configure AD-based authentication with the following ID mapping methods:

- RFC2307
- Automatic

- LDAP

## RFC2307 ID mapping

In the RFC2307 ID mapping method, the user and group IDs are stored and managed in the AD server and these IDs are used by the IBM Spectrum Scale system during file access. The RFC2307 ID mapping method is used when you want to have multiprotocol access. That is, you can have both NFS and SMB access over the same data.

## Automatic ID mapping

In the automatic ID mapping method, user ID and group ID are automatically generated and stored within the IBM Spectrum Scale system. When an external ID mapping server is not present in the environment or cannot be used, then this ID mapping method can be used. This method is typically used if you have SMB only access and do not plan to deploy multiprotocol access. That is, the AD-based authentication with automatic ID mapping is not used if you need to allow NFS and SMB access to the same data.

## LDAP ID mapping

In the LDAP mapping method, user ID and group ID are stored and managed in the LDAP server, and these IDs are used by the IBM Spectrum Scale system during file access. The LDAP ID mapping method is used when you want to have multiprotocol access. That is, you can have both NFS and SMB access over the same data.

## Setting up a range of ID maps that can be allotted to the users

You can optionally specify the pool of values from which the UIDs and GIDs are assigned by the IBM Spectrum Scale system to Active Directory users and groups. When a user or group is defined in AD, it is identified by a security identifier (SID), which includes a component that is called Relative Identifier (RID). The RID value depends on the number of users and groups in the Active Directory domain. The **--idmap-range** and **--idmap-range-size** parameters of the **mmuserauth service create** command specify the pool from which UIDs and GIDs are assigned by the IBM Spectrum Scale system to AD users and group of users.

The ID map range is defined between a minimum and maximum value. The default minimum value is 10000000 and the default maximum value is 299999999, and the default range size is 1000000. This allows for a maximum of 290 unique Active Directory domains.

The ID map range size specifies the total number of UIDs and GIDs that are assignable per domain. For example, if range is defined as 10000-20000, and range size is defined as 2000 (`--idmap-range 10000-20000:2000`), five domains can be mapped, each consisting of 2000 IDs. Ensure that range size is defined such that at least three domains can be mapped. The range size is identical for all AD domains that are configured by the IBM Spectrum Scale system. Choose an ID map range size that allows for the highest anticipated RID value among all of the anticipated AD users and group of users in all of the anticipated AD domains. Ensure that the range size value, when originally defined, takes into account the planned growth in the number of AD users and groups of users. The ID map range size cannot be changed after the IBM Spectrum Scale system is configured with Active Directory as the authentication server.

Whenever a user or user group from an AD domain accesses the IBM Spectrum Scale system, a range is allocated per domain. UID or GID for a user or user group is allocated depending upon this range and the RID of the user or user group. If RID of any user or group is greater than the range size, then that user or user group is mapped into extension ranges depending upon the number of available ranges. If the number of ranges (default value is 290) runs out, then mapping requests for a new user or user group (or new extension ranges for user and group that is already known) are ignored and thus that user and user group cannot access the data.

### Choosing range size

1. Determine the highest Active Directory RID that is currently assigned. You can use the **dcdiag** command at the command prompt of the operating system of the server that is hosting Active Directory to determine the value of the rIDNextRID attribute. For example:

   ```
   # dcdiag /s:IP_of_system_hosting_AD  /v /test:ridmanager
   ```

   Specifically,

   ```
   C:\Program Files\Support Tools>dcdiag /s:10.0.0.123 /v /test:ridmanager
   ```

   The following output is displayed:

   ```
   Starting test: RidManager
        * Available RID Pool for the Domain is 1600 to 1073741823
        * win2k8.pollux.com is the RID Master
        * DsBind with RID Master was succesRFC2307l
        * rIDAllocationPool is 1100 to 1599
        * rIDPreviousAllocationPool is 1100 to 1599
        * rIDNextRID: 1174
   ```

   In this example, the rIDNextRID value is 1174. Another way to determine the current value for rIDNextRID is to run an LDAP query on the following DN Path:

   ```
   CN=Rid Set,Cn=computername,ou=domain controllers,DC=domain,DC=COM
   ```

   If there is more than one domain controller serving the Active Directory domain, determine the highest RID among all of the domain controllers. Similarly, if there is more than one domain, determine the highest RID among all of the domains.
2. Estimate the expected number of users and groups that might be added in future, in addition to the current number of users and groups.
3. Add the highest RID determined in step 1 to the number of users and groups that were estimated in the previous step. The result is the estimate for the value of the range size.

## Considerations for changing the ID map range and range size

If the IBM Spectrum Scale system is configured to use AD-based authentication, only the maximum value of ID map range can be changed. All other changes to ID map range and size, except increasing the maximum value of ID map range require reconfiguration of authentication, which results in loss of access to data. For example, if you used the **--idmap-range** as 3000-10000 and **--idmap-range-size** as 2000, you can increase only the value 10000 to accommodate more users per domain, without having an impact on the access to the data.

To change the ID mapping of an existing AD-based authentication configuration, either to change the range minimum value, decrease the range maximum value, or change the range size, you must complete the following steps:

**Note:** The **mmuserauth service remove** command results in loss of access.

1. Submit the **mmuserauth service remove** command and do not specify the **--idmapdelete** option.
2. Submit the **mmuserauth service remove** command and do specify the **--idmapdelete** option.
3. Submit the **mmuserauth service create** command with the options and values that you want for the new Active Directory configuration.

   **Important:** If you do not perform the preceding three steps in sequence, results are unpredictable and can include complete loss of data access.

## Prerequisite for configuring AD-based authentication for file access

See "Integrating with AD server" on page 129 for more information on the prerequisites for integrating AD server with the IBM Spectrum Scale system.

You need to run the **mmuserauth service create** command with the following mandatory parameters to create AD based authentication for file access:

- --type ad

- --data-access-method file
- --servers <*comma-delimited server host names or IP addresses*>
- --netbios-name <*netBiosName*>
- --user-name <*admin-username*>
- --password <*admin-password*>. This is optional while entering the parameters but the system prompts you to enter the password when you run the command.
- --unixmap-domains <unixDomainMap>. This option is mandatory if RFC2307 ID mapping is used. For example, **--unixmap-domains** `DOMAINS(5000-20000)`. Specifies the Active Directory domains for which user ID and group ID should be fetched from the Active directory server ( RFC2307 schema attributes )
- --idmap-role `master | subordinate`. While using automatic ID mapping, in order to have same ID maps on systems sharing Active File Manager (AFM) relationship, you need to export the ID mappings from the system whose ID map role is master to the system whose ID map role is subordinate.

See the **mmuserauth service create** command for more information on each parameter.

### Prerequisites for configuring AD with RFC2307

The following prerequisites are specific to AD with RFC2307 configuration:

- RFC2307 schema is extended on the AD and all UNIX attributes (including UID and GID) are populated.
- If a trusted domain is configured with ID mapping from RFC2307, the trusted domain must have two-way trust with the host domain, which is the Active Directory domain that is configured for use with the IBM Spectrum Scale system. For example, assume that there are three domains in trusted relationship, X, Y, Z, and that the IBM Spectrum Scale system is configured with domain X as the host domain. If RFC2307 ID mappings are required for domains Y and Z, domains Y and Z must each have a two-way trust with the domain X. `X <-> Y ; X <-> Z`.
- User and group in the Active Directory domain, configured with ID mapping from RFC2307, must have a valid UID and a valid GID assigned to enable access to IBM Spectrum Scale system exports. The UID and GID number that is assigned must be within the ID map range that is specified in the **mmuserauth service create** command. Any users or groups from this domain that do not have UID or GID attributes configured are denied access.

**Note:** The primary Windows group that is assigned to an AD user must have a valid GID assigned within the specified ID mapping range. The primary Windows group is usually located in the Member Of tab in the user's properties. The primary Windows group is different from the UNIX primary group, which is listed in the UNIX Attributes tab. A user is denied access if that user's Windows primary group does not have a valid GID assigned. The UNIX primary group attribute is ignored.

In case of a mutual trust setup between two independent AD domains, DNS forwarding must be configured between the two trust.

## Configuring AD-based authentication with automatic ID mapping

When the IBM Spectrum Scale system is configured for AD-based authentication, automatic ID mapping method can be used to create UID or GID of a user or group respectively. The ID maps are stored within the IBM Spectrum Scale system.

The following provides an example of how to configure an IBM Spectrum Scale system with Active Directory and automatic ID mapping.

1. Issue the **mmuserauth service create** command as shown in the following example:

   ```
   # mmuserauth service create --type ad --data-access-method file --netbios-name
   ess --user-name administrator --idmap-role master --servers myADserver
   --password Passw0rd --idmap-range-size 1000000 --idmap-range 10000000-299999999
   ```

   The system displays the following output:

```
File Authentication configuration completed successfully.
```

2. Verify the authentication configuration by issuing the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_NFS_KERBEROS     false
SERVERS                 myADserver
USER_NAME               administrator
NETBIOS_NAME            ess
IDMAP_ROLE              master
IDMAP_RANGE             10000000-299999999
IDMAP_RANGE_SIZE        1000000
UNIXMAP_DOMAINS         none

OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

3. Verify the user resolution on the system:

```
# id "DOMAIN\\user1"
uid=12001172(DOMAIN\user1) gid=12001174(DOMAIN\group1) groups=12001174
(DOMAIN\group1),12001172(DOMAIN\user1),12000513(DOMAIN\domain users),
11000545(BUILTIN\users)
```

## Prerequisite for configuring Kerberos-based SMB access

The following requirements must be met to configure IBM Spectrum Scale for Kerberized SMB access:

- The time must be synchronized across the KDC server, the IBM Spectrum Scale cluster, and the SMB clients, or else access to an SMB share could be denied.
- In MIT KDC configurations for the SMB services, the service principal name must use the NetBIOS name and the realm name. For example, if the NetBIOS name is FOO and the realm is KDC.COM, the service principal name should be cifs/foo@KDC.COM. The NetBIOS name is the value specified for the option --netbios_name in the **mmuserauth** command. The realm may be discovered from the value stored for Alt_Name returned from the command: wbinfo -D <domain>.
- The clients should use only the NetBIOS name when accessing an SMB share. Using any other name or IP address might either cause a failure to connect or fallback to NTLM authentication.
- With Active Directory KDC, you can use DNS alias (CNAME) for Kerberized SMB access. To use the alias, you must register the DNS alias (CNAME) record for the NetBIOS name (system account name) using the SetSPN tool available on Active Directory server. For example, if the NetBIOS name is FOO and the DNS alias is BAR, use the SetSPN tool from the command prompt of the Active Directory server to register the record, "setspn -A cifs/BAR FOO". Not registering the DNS alias record for the NetBIOS name might cause access to the SMB shares to be denied with the error code, KDC_ERR_S_SPRINCIPAL_UNKNOWN.
- On Linux clients, to use Kerberized SMB access for IBM Spectrum Scale configured with MIT KDC, you must at least have the 3.5.9 version of Samba client installed. The Linux clients having an older Samba client might encounter the following error, while trying to access SMB shares:

```
ads_krb5_mk_req: krb5_get_credentials failed for foo$@KDC.COM (Server not found in Kerberos database)
 cli_session_setup_kerberos: spnego_gen_negTokenTarg failed: Server not found in Kerberos database
```

  To determine if a client has authenticated via Kerberos, either verify at the client or collect a protocol trace:

```
mmprotocoltrace start smb -c x.x.x.x
```

  Replace x.x.x.x with the IP address of the client system access IBM Spectrum Scale to be verified.

  Access the IBM Spectrum Scale SMB service from that client.

  Then, issue the command:

```
mmprotocoltrace stop smb
```

Extract the compressed trace files and look for the file ending with smbd.log. If that file contains an entry similar to "Kerberos ticket principal name is..." then Kerberosis being used.

**Note:** It is not recommended to run for extended periods of time at log levels higher than 1 as this could impact performance.

## Configuring AD-based authentication with RFC2307 ID mapping

You can configure IBM Spectrum Scale system authentication with Active Directory (AD) and RFC2307 and Active Directory (AD) with Kerberos and RFC2307 ID mapping. In these authentication methods, use Active Directory to store user credentials and RFC2307 server to store UIDs and GIDs. This is useful when you are planning to use any pre-existing UNIX client or NFS and SMB protocols for data access with the AFM feature of the IBM Spectrum Scale system. If you use AD-based authentication and the ID maps are not configured with RFC2307, the IBM Spectrum Scale system uses the automatic ID mappings by default.

**The following provides an example of configuring AD with RFC2307 ID mapping:**

1. Submit the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create  --type ad --data-access-method file --netbios-name
ess --user-name administrator --idmap-role master --servers myADserver
--password Passw0rd --idmap-range-size 1000000 --idmap-range 10000000-299999999
--unixmap-domains 'DOMAIN(5000-20000)'
```

   The system displays the following output:

```
File authentication configuration completed successfully.
```

2. Issue the **mmuserauth service list** to verify the authentication configuration as shown in the following example:

```
# mmuserauth service list
```

   The system displays the following output:

```
FILE access configuration : AD
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_NFS_KERBEROS     false
SERVERS                 myADserver
USER_NAME               administrator
NETBIOS_NAME            ess
IDMAP_ROLE              master
IDMAP_RANGE             10000000-299999999
IDMAP_RANGE_SIZE        1000000
UNIXMAP_DOMAINS         DOMAIN(5000-20000)
LDAPMAP_DOMAINS         none

OBJECT access not configured
PARAMETERS              VALUES
```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from RFC2307 attributes on the AD server.

```
# id DOMAIN\\administrator
uid=10002(DOMAIN\administrator) gid=10000(DOMAIN\domain users)
groups=10000(DOMAIN\domain users
```

**Configuring AD using Kerberos with RFC2307 ID mapping:**

1. Submit the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create  --data-access-method file --type ad --netbios-name
kknode_v42 --servers myADserver --user-name administrator --password Passw0rd --idmap-role master
--enable-nfs-kerberos --unixmap-domains "DOMAIN(10000-200000)"
```

   The system displays the following output:

```
File authentication configuration completed successfully.
```

2. Issue the **mmuserauth service list** to verify the authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : AD
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_NFS_KERBEROS     true
SERVERS                 myADserver
USER_NAME               administrator
NETBIOS_NAME            kknode_v42
IDMAP_ROLE              master
IDMAP_RANGE             10000000-299999999
IDMAP_RANGE_SIZE        1000000
UNIXMAP_DOMAINS         DOMAIN(1000-200000)
LDAPMAP_DOMAINS         none

OBJECT access not configured
PARAMETERS              VALUES
```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from RFC2307 attributes on the AD server.

```
# id DOMAIN\\administrator
uid=10002(DOMAIN\administrator) gid=40000(DOMAIN\domain users)
groups=11000545(BUILTIN\users),11000544 (BUILTIN\administrators)
```

**Best practices for configuring AD with RFC2307 as the authentication method:**

It is recommended to adhere to the following best practices if you configure AD with RFC2307 as the authentication method:

- Remove any internal ID mappings present in the system before you configure AD with RFC2307. Otherwise, the system might detect the internal ID mappings instead of the RFC2307 ID mapping and abort the operation with an error message. In such situations, you are expected to clean up the entire authentication and ID mapping by using the **mmuserauth service remove** and **mmuserauth service remove --idmapdelete** command and then reconfigure AD authentication and RFC2307 ID mapping.
- If data is already present on the system, a complete removal of the authentication and ID mapping can cause permanent loss of data access.
- Using UIDs and GIDs greater than 1000 can avoid an overlap of IDs used by end users, administrative users, and operating system component users of the IBM Spectrum Scale system.

You can use AD-based authentication and RFC2307 ID mapping if you want to use the AFM feature of the IBM Spectrum Scale system.

**Limitations of the mmuserauth service create command while configuring AD with RFC2307:**

The **mmuserauth service create** command that is used to configure authentication has the following limitations:

- The **mmuserauth service create** command does not check the two-way trust between the host domain and the RFC2307 domain that is required for ID mapping services to function properly. The customer is responsible for configuring the two-way trust relationship between these domains.
- The customer is responsible for installing RFC2307 on the desired AD server, and for assigning UIDs to users and GIDs to groups. The command does not return an error if RFC2307 is not installed, or if a UID or GID is not assigned.

### Configuring AD-based authentication with LDAP ID mapping

AD authentication with LDAP ID mapping provides a way for IBM Spectrum Scale to read ID mappings from an LDAP server as defined in RFC 2307. The LDAP server must be a stand-alone LDAP server. Mappings must be provided in advance by the administrator by creating the user accounts in the AD server and the posixAccount and posixGroup objects in the LDAP server. The names in the AD server and in the LDAP server have to be the same. This ID mapping approach allows the continued use of existing LDAP authentication servers that store records in the RFC2307 format. The group memberships defined in the AD server are also be honored in the system.

In the following example, AD is configured with LDAP ID mapping.

1. Submit the **mmuserauth service create** command as shown in the following example:

   ```
   mmuserauth service create --data-access-method file --type ad --servers myADserver
   --user-name administrator --password Passw0rd --netbios-name specscale
   --idmap-role master --ldapmap-domains "DOMAIN1(type=stand-alone:range=1000-100000
   :ldap_srv=myLDAPserver:usr_dn=ou=People,dc=example,dc=com:grp_dn=ou=Groups,dc=example,
   dc=com:bind_dn=cn=manager,dc=example,dc=com:bind_dn_pwd=password)
   ```

   **Note:** The **bind_dn_password** cannot contain the following special characters: semicolon (;), colon (:), opening brace '( ', or closing brace ')'.
   The system displays the following output:

   ```
   File authentication configuration completed successfully.
   ```

2. Issue the **mmuserauth service list** to verify the authentication configuration as shown in the following example:

   ```
   # mmuserauth service list
   ```

   The system displays the following output:

   ```
   FILE access configuration : AD
   PARAMETERS VALUES
   -------------------------------------------------
   ENABLE_NFS_KERBEROS false
   SERVERS myADserver
   USER_NAME administrator
   NETBIOS_NAME specscale
   IDMAP_ROLE master
   IDMAP_RANGE 10000000-299999999
   IDMAP_RANGE_SIZE 1000000
   UNIXMAP_DOMAINS none
   LDAPMAP_DOMAINS DOMAIN1(type=stand-alone: range=1000-100000:
   ldap_srv=myLDAPserver:usr_dn=ou=People,dc=example,dc=com:
   grp_dn=ou=Groups,dc=example,dc=com:bind_dn=cn-manager,dc=example,dc=com)
   ```

3. Verify the user name resolution on the system. Confirm that the resolution is showing IDs that are pulled from LDAP attributes on the AD server.

   ```
   # id DOMAIN\\administrator
   uid=10002(DOMAIN\administrator) gid=10000(DOMAIN\domain users)
   groups=10000(DOMAIN\domain users
   ```

## Configuring LDAP-based authentication for file access

Using LDAP-based authentication can be useful when you use an external LDAP server to store user information and user passwords. In this authentication method, you can use LDAP as the authentication as well as the ID mapping server for both NFS and SMB. Appropriate SMB schema needs to be uploaded in the LDAP if you plan to have SMB access.

Based on the level of security, the following configurations are possible:

- LDAP with TLS
- LDAP with Kerberos
- LDAP with TLS and Kerberos

- LDAP

Using LDAP with TLS secures the communication between the IBM Spectrum Scale system and the LDAP server, assuming that the LDAP server is configured for TLS.

You can use LDAP with Kerberos for higher security reasons. Kerberos is a network authentication protocol that provides secured communication by ensuring passwords are not sent over the network to the system. LDAP with Kerberos is typically used where an MIT KDC infrastructure exists and you are using it for various Kerberized application or if you want to have NFS and SMB with Kerberized access for higher security reasons.

The LDAP server might need to handle the login requests and ID mapping requests from the client that uses SMB protocol. Usually, the ID mapping requests are cached and they do not contribute to the load on the LDAP server unless the ID mapping cache is cleared due to a maintenance action. If the LDAP server cannot handle the load or a high number of connections, then the response to the login requests is slow or it might time out. In such cases, users need to retry their login requests.

It is assumed that LDAP server is set up with the required schemas installed in it to handle the authentication and ID mapping requests. If you need to support SMB data access, LDAP schema must be extended to enable storing of additional attributes such as SID, Windows password hash to the POSIX user object.

**Note:** The IBM Spectrum Scale system must not be configured with any authentication method before using LDAP as the authentication system for file access.

See "Integrating with LDAP server" on page 130 for more information on the prerequisites for integrating LDAP server with the IBM Spectrum Scale system.

## Configuring LDAP with TLS for file access

You can configure LDAP with TLS as the authentication method for file access. Using TLS with LDAP helps you to have a secure communication channel between the IBM Spectrum Scale system and LDAP server.

In the following example, LDAP is configured with TLS as the authentication method for file access.

1. Ensure that the CA certificate for LDAP server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem`; specifically, on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```
# stat /var/mmfs/tmp/ldap_cacert.pem
File: ⌂/var/mmfs/tmp/ldap_cacert.pem⌂
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. Issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-server-tls
```

   The system displays the following output:

```
File authentication configuration completed successfully.
```

3. Issue the **mmuserauth service list** command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_SERVER_TLS       true
ENABLE_KERBEROS         false
USER_NAME               cn=manager,dc=example,dc=com
SERVERS                 myLDAPserver
NETBIOS_NAME            ess
BASE_DN                 dc=example,dc=com
USER_DN                 none
GROUP_DN                none
NETGROUP_DN             none
USER_OBJECTCLASS        posixAccount
GROUP_OBJECTCLASS       posixGroup
USER_NAME_ATTRIB        cn
USER_ID_ATTRIB          uid
KERBEROS_SERVER         none
KERBEROS_REALM          none

OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

4. Verify the user resolution on system present in LDAP:

```
# id ldapuser2
uid=1001(ldapuser2) gid=1001(ldapuser2) groups=1001(ldapuser2)
```

## Configuring LDAP with Kerberos for file access

You can configure LDAP with Kerberos as the authentication method for file access. Using Kerberos with LDAP provides more security for the communication channel between the IBM Spectrum Scale system and LDAP server.

Example for configuring LDAP with Kerberos as the authentication method for file access.

1. Ensure that the keytab file is also placed under the `/var/mmfs/tmp` directory with the name as `krb5.keytab` on the node where the command is run. Perform validation of keytab file availability with desired name at required location:

```
# stat /var/mmfs/tmp/krb5.keytab
File: ⌂/var/mmfs/tmp/krb5.keytab⌂
Size: 502 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169898 Links: 1
Access: (0600/-rw-------) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 14:31:18.244837381 +0530
Modify: 2015-01-23 12:45:05.475837381 +0530
Change: 2015-01-23 12:45:05.476837381 +0530
Birth: -
```

2. Issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers myLDAPserver --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret
--netbios-name ess --enable-kerberos --kerberos-server myKerbersosServer
--kerberos-realm example.com
```

The system displays the following output:

```
File authentication configuration completed successfully.
```

3. Issue the **mmuserauth service list** command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_SERVER_TLS       false
ENABLE_KERBEROS         true
USER_NAME               cn=manager,dc=example,dc=com
SERVERS                 myLDAPserver
NETBIOS_NAME            ess
BASE_DN                 dc=example,dc=com
USER_DN                 none
GROUP_DN                none
NETGROUP_DN             none
USER_OBJECTCLASS        posixAccount
GROUP_OBJECTCLASS       posixGroup
USER_NAME_ATTRIB        cn
USER_ID_ATTRIB          uid
KERBEROS_SERVER         myKerbersosServer
KERBEROS_REALM          example.com

OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

## Configuring LDAP with TLS and Kerberos for file access

You can configure LDAP with TLS and Kerberos as the authentication method for file access. Using Kerberos and TLS with LDAP provides maximum security for the communication channel between the IBM Spectrum Scale system and the LDAP server.

Provides an example on how to configure LDAP with TLS and Kerberos as the authentication method for file access.

1. Ensure that the CA certificate for LDAP server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem`; specifically, on the protocol node where the command is run. Perform validation of CA cert availability with desired name at the required location as shown in the following example:

   ```
   # stat /var/mmfs/tmp/ldap_cacert.pem
   File: ⌂/var/mmfs/tmp/ldap_cacert.pem⌂
   Size: 2130 Blocks: 8 IO Block: 4096 regular file
   Device: fd00h/64768d Inode: 103169903 Links: 1
   Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
   Context: unconfined_u:object_r:user_tmp_t:s0
   Access: 2015-01-23 12:37:34.088837381 +0530
   Modify: 2015-01-23 12:16:24.438837381 +0530
   Change: 2015-01-23 12:16:24.438837381 +0530
   ```

2. Ensure that the keytab file is placed under `/var/mmfs/tmp` directory name as `krb5.keytab` specifically on the node where the command is run. Perform validation of keytab file availability with desired name at the required location:

   ```
   # stat /var/mmfs/tmp/krb5.keytab
   File: ⌂/var/mmfs/tmp/krb5.keytab⌂
   Size: 502 Blocks: 8 IO Block: 4096 regular file
   Device: fd00h/64768d Inode: 103169898 Links: 1
   Access: (0600/-rw-------) Uid: ( 0/ root) Gid: ( 0/ root)
   Context: unconfined_u:object_r:user_tmp_t:s0
   Access: 2015-01-23 14:31:18.244837381 +0530
   Modify: 2015-01-23 12:45:05.475837381 +0530
   Change: 2015-01-23 12:45:05.476837381 +0530
   Birth: -
   ```

3. Issue the **mmuserauth service create** command as shown in the following example:

   ```
   # mmuserauth service create --type ldap --data-access-method file
   --servers myLDAPserver --base-dn dc=example,dc=com
   --user-name cn=manager,dc=example,dc=com --password secret
   --netbios-name ess --enable-server-tls --enable-kerberos
   --kerberos-server myKerbersosServer --kerberos-realm example.com
   ```

The system displays the following output:

```
File authentication configuration completed successfully.
```

4. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_SERVER_TLS       true
ENABLE_KERBEROS         true
USER_NAME               cn=manager,dc=example,dc=com
SERVERS                 myLDAPserver
NETBIOS_NAME            ess
BASE_DN                 dc=example,dc=com
USER_DN                 none
GROUP_DN                none
NETGROUP_DN             none
USER_OBJECTCLASS        posixAccount
GROUP_OBJECTCLASS       posixGroup
USER_NAME_ATTRIB        cn
USER_ID_ATTRIB          uid
KERBEROS_SERVER         myKerbersosServer
KERBEROS_REALM          example.com

OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

5. Verify the user resolution on the system:

```
# id ldapuser3
uid=1002(ldapuser3) gid=1002(ldapuser3) groups=1002(ldapuser3)
```

## Configuring LDAP without TLS and Kerberos for file access

You can configure LDAP without TLS or Kerberos for file access. But this method is less secured compared to LDAP with TLS, LDAP with TLS and Kerberos, and LDAP with Kerberos configurations.

Provides an example on how to configure LDAP without TLS and Kerberos as the authentication method for file access.

1. Issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method file
--servers 192.0.2.18 --base-dn dc=example,dc=com
--user-name cn=manager,dc=example,dc=com --password secret --netbios-name ess
```

The system displays the following output:

```
File Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access configuration : LDAP
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_SERVER_TLS       false
ENABLE_KERBEROS         false
USER_NAME               cn=manager,dc=example,dc=com
SERVERS                 192.0.2.18
NETBIOS_NAME            ess
```

```
BASE_DN                 dc=example,dc=com
USER_DN                 none
GROUP_DN                none
NETGROUP_DN             none
USER_OBJECTCLASS        posixAccount
GROUP_OBJECTCLASS       posixGroup
USER_NAME_ATTRIB        cn
USER_ID_ATTRIB          uid
KERBEROS_SERVER         none
KERBEROS_REALM          none

OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

## Configuring NIS-based authentication

The Network Information Service (NIS)-based authentication is useful in NFS only environment where NIS acts as an ID mapping server and also used for netgroups. When file access is configured with NIS, SMB access cannot be enabled.

Ensure that you have the following details before you start NIS-based authentication:

- NIS server name. This is case-specific
- IP address or host name of the NIS server
- Primary DNS is added in the /etc/resolv.conf file on all the protocol nodes. It resolves the authentication server system with which the IBM Spectrum Scale system is configured. The manual changes done to the configuration files might get overwritten by the Operating System's network manager. So, ensure that the DNS configuration is persistent even after you restart the system. For more information on the circumstances where the configuration files are overwritten, refer the corresponding Operating System documentation.

You need to run the **mmuserauth service create** command with the following mandatory parameters to configure NIS as the authentication method:

- --type nis
- --data-access-method file
- --domain `domainName`
- --servers `comma-delimited IP address or host name`

For more information on each parameter, see the **mmuserauth service create** command.

Provides an example on how to configure NIS as the authentication method for file access.

1. Issue the **mmuserauth service create** command as shown in the following example:

   ```
   # mmuserauth service create --type nis --data-access-method file
   --servers myNISserver --domain nisdomain3
   ```

   The system displays the following output:

   ```
   File Authentication configuration completed successfully.
   ```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

   ```
   # mmuserauth service list
   ```

   The system displays the following output:

   ```
   FILE access configuration : NIS
   PARAMETERS              VALUES
   -------------------------------------------------
   SERVERS                 myNISserver
   DOMAIN                  nisdomain3
   ```

```
OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------.
```

## Managing user-defined authentication

In the user-defined mode of authentication, the user is free to select the authentication and ID mapping methods of their choice. It is the responsibility of the administrator of the client system to manage the authentication and ID mapping for file (NFS and SMB) and object access to the IBM Spectrum Scale system.

The IBM Spectrum Scale system administrators are not allowed use any of the GPFS commands to manage authentication. It is important for the end user to be aware of the limitations, if any, of the authentication and ID mapping scheme that will be implemented after configuring the user-defined mode of authentication.

The user-defined mode is appropriate in the following circumstances:

- The client already has protocol deployments either on GPFS installations or on different systems and is planning to move to using the protocol stack on the IBM Spectrum Scale system. The client wants to replicate the current authentication and ID mapping configuration. In this case, the client system administrator must be familiar with the required configuration settings that will be applied to the system.
- If the end user wants an authentication method that is not supported by the IBM Spectrum Scale system.

**Note:** If the end user wants to configure the authentication methods that are supported by the IBM Spectrum Scale system, it is highly recommended to configure the authentication and ID mapping methods by using the **mmuserauth** command instead of opting for the user-defined method of authentication.

The IBM Spectrum Scale system administrator needs to specify that the user-defined mode of authentication is used by using the **--type userdefined** option in the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type userdefined --data-access-method file
File Authentication configuration completed successfully.
```

Submit the **mmuserauth service list** command to see the current authentication configuration as shown in the following example:

```
# mmuserauth service list
FILE access configuration : USERDEFINED
PARAMETERS              VALUES
-------------------------------------------------
OBJECT access not configured
PARAMETERS              VALUES
-------------------------------------------------
```

Typically, user-defined authentication is used when existing GPFS customers are already using GPFS with NFS and do not want to alter the authentication that is already configured on these systems. You can configure user-defined authentication for both object and file access or for object or file alone.

**Note:** Authorization depends upon authentication and ID mapping that is configured with the system. That is, the ACL control on exports, files, and directories depend on the authentication method that is configured.

Ensure the following while using the user-defined mode of authentication for file access:

- Ensure that the authentication server and ID mapping server are always reachable from all the protocol nodes. For example, if NIS is configured as the ID mapping server, you can use the 'ypwhich' command to ensure that NIS is configured and reachable from all the protocol nodes. Similarly, if LDAP is configured as authentication and ID mapping server, you can bind to the LDAP server from all protocol nodes to monitor if the LDAP server is reachable from all protocol nodes.
- Ensure that the implemented authentication and ID mapping configuration is always consistent across all the protocol nodes. This requires that the authentication server and ID mapping server are manually maintained and monitored by the administrator. The administrator must also ensure that the configuration files are not overwritten due to node restart and other similar events.
- Ensure that the implemented authentication and ID mapping-related daemons and processes across the protocol nodes are always up and running.
- The users or groups, accessing the IBM Spectrum Scale system over NFS and SMB protocols must resolve to a unique UID and GID respectively on all protocol nodes, especially in implementations where different servers are used for authentication and ID mapping. The name that is registered in ID mapping server for user and group must be checked for resolution.

  For example:

  ```
  # id fileuser
  uid=1234(fileuser) gid=5678(filegroup) groups=5678(filegroup)
  ```

  **Note:** However, there are some use cases where only NFSV3 based access to the IBM Spectrum Scale system is used. In such cases, the user and group IDs are obtained from the NFS client and there is no ID mapping setting is configured on the protocol nodes.
- If the IBM Spectrum Scale system is configured for multiprotocol support (that is, the same data is accessed through both NFS and SMB protocols), ensure that the IDs of users and groups are consistent across the NFS clients and SMB clients and that they resolve uniquely on the protocol nodes.
- Ensure that there is no conflict of UID and GID across users and groups that are accessing the system. This must be strictly enforced, especially in multiprotocol-based access deployments.
- Ensure that the Kerberos configuration files, placed on all protocol nodes, are in synchronization with each other. Ensure that the clients and the IBM Spectrum Scale system are part of the same Kerberos realm or trusted realm.
- While deploying two or more IBM Spectrum Scale clusters, ensure that the ID mapping is consistent in cases where you want to use IBM Spectrum Scale features like AFM, AFM-DR, and asynchronous replication of data.

The following table provides an overview of the authentication requirements for each file access protocol. Refer this table when you plan to use user-defined mode as the authentication method.

*Table 18. Authentication requirements for each file access protocol.*

| File access protocol | Requirements |
|---|---|
| NFSV3 | In scenarios where user name and group name are expected to be used to native GPFS commands (for example, setting data ownership, listing user or group quota), the IBM Spectrum Scale system must be able to resolve the UID and GID to user name and group name and vice versa, consistently across all the protocol nodes. **Note:** However, there are some use cases where only the NFSv3 based access to the IBM Spectrum Scale system is used. In such cases, the user and group IDs are coming from the NFS client and there is no ID mapping setting is configured on the protocol nodes. |

*Table 18. Authentication requirements for each file access protocol. (continued)*

| File access protocol | Requirements |
|---|---|
| Kerberos NFSV3 | Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.<br><br>Ensure that the time is synchronized on the NFS server, NFS clients, and Kerberos server.<br>**Note:** User names and group names are case-sensitive. |
| NFSV4 | Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.<br><br>Domain name must be specified in the `etc/idmapd.conf` file and it must be the same on both the NFS server and NFS clients.<br>**Note:** User names and group names are case-sensitive. |
| Kerberos NFS V4 | Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.<br><br>Ensure that the time is synchronized on the NFS server, NFS clients, and Kerberos server.<br><br>Domain name and local-realms must be specified in the `etc/idmapd.conf` file and it must be the same on both the NFS server and NFS clients.<br><br>The value of "local-realms" takes the value of Kerberos realm with which the IBM Spectrum Scale system protocol nodes are configured. |
| SMB | Ensure that the user name and group name that are used to access data consistently resolve to same UID and GID across all protocol nodes and NFS clients.<br><br>While integrating with non-windows server, ensure that the samba attributes are populated on the directory server for every user and group that are planning to access the IBM Spectrum Scale system. Special care must be taken to match the samba domain SIDs.<br><br>For Kerberized SMB access, ensure that time is synchronized the SMB server, SMB client, and Kerberos server. |

The user-defined mode for object authentication integrates IBM Spectrum Scale Object Storage with the externally hosted keystone server. Ensure the following while using the user-defined mode of authentication for object access:

- Integration with external keystone server is supported over http and https.
- The specified object user must be defined while enabling and configuring object in the external keystone server.
- The 'service' tenant must be defined in external keystone server.
- The 'admin' role must be defined in the external keystone server.
- Ensure that the specified swift user has 'admin' role in 'service' tenant.
- Object storage service endpoints must be correctly defined in the external keystone server.

  For example, the external keystone server must contain the following endpoint for object-store:

```
# openstack endpoint list
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
| ID         | Region | Service Name | Service Type | Enabled | Interface | URL                                                             |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
| c36e..9da5 | None   | keystone     | identity     | True    | public    | http://specscaleswift.example.com:5000/                         |
| f4d6..b040 | None   | keystone     | identity     | True    | internal  | http://specscaleswift.example.com:35357/                        |
| d390..0bf6 | None   | keystone     | identity     | True    | admin     | http://specscaleswift.example.com:35357/                        |
| 2e63..f023 | None   | swift        | object-store | True    | public    | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s    |
| cd37..9597 | None   | swift        | object-store | True    | internal  | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s    |
| a349..58ef | None   | swift        | object-store | True    | admin     | http://specscaleswift.example.com:8080                          |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
```

If the object authentication is set to 'user-defined' and an IP address/port number is set in the proxy server configuration for keystone authentication, then that IP address will be checked using a simple http(s) request. If the request fails, then the AUTH_OBJ state will be set to "degraded" and an 'external keystone URL failure' event will be logged. This will not cause the node to be flagged as bad nor will it cause any public IP movement.

- Issue the following command:

  ```
  mmces state show
  ```

  The system displays output similar to this:

  ```
  NODE                       AUTH      AUTH_OBJ   NETWORK  NFS       OBJ       SMB       CES
  spectrum-31.localnet.com   DISABLED  DEGRADED   HEALTHY  DISABLED  DEGRADED  HEALTHY   DEGRADED
  ```

- Issue the following command:

  ```
  mmces events list
  ```

  The system displays output similar to this:

  ```
  NODE                     TIMESTAMP                          EVENT NAME    SEVERITY  DETAILS
  spectrum-31.localnet.com 2015-10-18 18:23:05.386336--1:-1CEST ks_url_exfail WARNING   Keystone request failed
                                                    using http://10.11.0.1:35357/v2.0
  ```

## Configuring authentication for object access

You can use the following authentication methods for object access:

- Active Directory (AD)
- LDAP
- Local authentication

The AD-based and LDAP-based authentication methods use an external AD and LDAP server respectively to manage the authentication. Local authentication is handled by a Keystone server that resides within the IBM Spectrum Scale system.

The IBM Spectrum Scale system installation process configures Keystone server that is required for object access. By default the IBM Spectrum Scale installation process configures object authentication with a local Keystone authentication method. If you have an existing Keystone server that you want to use, specify that it be used for authentication.

Before you configure object authentication method, ensure that the Keystone Identity service is properly configured.

**Note:** Before you configure an authentication method for object access, ensure that all protocol nodes have CES IP addresses assigned and you are issuing the authentication configuration command from the protocol node that has one or more CES IP addresses assigned to it.

Before you start manually configuring authentication method for object access, ensure that the `openldap-clients` RPM is installed.

On each protocol node, issue the following command: **`yum install openldap-clients`**.

**Note:** This step is required only when the authentication type is AD/LDAP.

The mapping between user, role, and tenant is stored in the Keystone database. If you switch from one authentication type to another you must delete the existing mapping definitions by running the **mmuserauth service remove --idmapdelete** command.

**Note:**

It is recommended to run the **mmuserauth service check** command as follows after configuring object authentication using the **mmuserauth service create** command:

```
mmuserauth service check --data-access-method object -N cesNodes
```

If the **mmuserauth service check** command reports that any certificate file is missing on any of the nodes, then run the following command:

```
mmuserauth service check --data-access-method object -N cesNodes --rectify
```

For more information about **mmuserauth service check**, see the topic *mmuserauth command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Configuring local authentication for object access

Object access can be configured with the Keystone server that is available in the IBM Spectrum Scale system. In this mode, Keystone stores the identity and assignment information locally in its database.

The local authentication method is useful when you want to create and maintain a separate set of users for only object access. These users cannot use the local authentication credentials for accessing file data that is hosted through NFS and SMB protocols. If you want to allow a user to access both file and object, use an external authentication server such as AD or LDAP to manage user accounts and authentication requests.

**Note:** File and object authentication must be configured with individual invocations of the **mmuserauth** command, even if the authentication server is the same.

You need to use the **mmuserauth service create** command with the following mandatory parameters to configure local authentication for object access:

- --type local
- --data-access-method `object`
- --ks-dns-name `keystoneDNSName`
- --ks-admin-user `keystoneAdminName`
- --ks-admin-pwd `keystoneAdminPwd`. If not provided, the system prompts to enter the password during the command execution.

For more information on each parameter, see the **mmuserauth service create** command.

1. To configure local authentication for object access, issue **mmuserauth service create** command as shown in the following example:

   ```
   # mmuserauth service create --data-access-method object --type local
   --ks-dns-name c40bbc2xn3 --ks-admin-user admin --ks-admin-pwd Passw0rd
   ```

   The system displays the following output:

   ```
   Object configuration with local (Database) as identity backend is completed
   successfully.
   Object Authentication configuration completed successfully.
   ```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

   ```
   # mmuserauth service list
   ```

The system displays the following output:

```
FILE access not configured
PARAMETERS               VALUES
-------------------------------------------------

OBJECT access configuration : LOCAL
PARAMETERS               VALUES
-------------------------------------------------
ENABLE_KS_SSL            false
ENABLE_KS_CASIGNING      false
KS_ADMIN_USER            admin
```

### Configuring local authentication with SSL for object access

Use the following steps to configure object access with the Keystone server that is available in the IBM Spectrum Scale system with SSL enabled.

1. Obtain certificates from the Certification Authority (CA) and place them at the following location on the current node from where the **mmuserauth service create** command is being executed.

   ```
   /var/mmfs/tmp/ssl_cert.pem
   /var/mmfs/tmp/ssl_key.pem
   /var/mmfs/tmp/ssl_cacert.pem
   ```

   **Note:**
   - Self-signed certificates can be used for testing and demonstration purposes. However, the use of externally signed certificates is strongly recommended for production environments.
   - The name in the SSL certificate must match the Keystone endpoint name.
2. Remove existing local authentication for object access as follows.

   ```
   mmuserauth service remove --data-access-method object
   ```
3. Configure local authentication with SSL for object access as follows.

   ```
   mmuserauth service create --data-access-method object --type local --enable-ks-ssl
   ```

Local authentication is now configured for object access with SSL enabled.

To disable SSL and configure local authentication for object access again, use the following steps.

4. Remove existing local authentication for object access as follows.

   ```
   mmuserauth service remove --data-access-method object
   ```

   If you are also changing authentication type, remove authentication and ID mappings by using the following commands in sequence.

   ```
   mmuserauth service remove --data-access-method object
   mmuserauth service remove --data-access-method object --idmapdelete
   ```
5. Configure local authentication without SSL for object access as follows.

   ```
   mmuserauth service create --data-access-method object --type local
   ```

## Configuring an AD-based authentication for object access

You can configure Keystone with an external AD server as the authentication back-end so that AD users can access the object store by using their AD credentials. The same AD server can be used for both object access and file access.

The AD server is set up to handle the authentication requests. AD is used as an LDAP server. Unlike file access, multiple AD domains are not supported.

### Prerequisites

Ensure that you have the following details before you start AD-based authentication configuration:

- AD server details such as IP address or host name, user name, user password, base dn, and user dn.

- If you want to configure TLS with AD for secure communication between Keystone and AD, you need to place the CA certificate that is used for signing the AD server setup for TLS under the following directory of the node on which the **mmuserauth service create** command is run:
  - `/var/mmfs/tmp/ldap_cacert.pem`
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.

See "Integrating with AD server" on page 129 for more information on the prerequisites for integrating AD server with the IBM Spectrum Scale system.

The following parameters must be used with **mmuserauth service create** command to configure AD-based authentication for object access:

- **--type** ad
- **--data-access-method** object
- **--servers** IP address or host name of AD. All user lookups by Keystone are done only against this server. If multiple servers are specified, only the first server is used and the rest are ignored.
- **--base-dn** ldapBase
- { **--enable-anonymous-bind** | **--user-name BindDN --password BindPwd**} You need to mention either anonymous bind or either –user-name or –password.
- **--enable-server-tls**, if TLS needs to be enabled
- **--user-dn** ldapUserSuffix. LDAP container from where users are looked up.
- **--ks-dns-name** keystoneDNSName
- **--ks-admin-user** keystoneAdminUser from AD
- **--enable-ks-ssl**, if SSL needs to be enabled. You need to have another set of certificates that are placed in standard directory.
- **--enable-ks-casigning**, if you want to use external CA signed certificate for token signing
- **--ks-swift-user** Swift_Service_User from AD
- **--ks-swift-pwd** Swift_Service_User's Password from AD

For more information on each parameter, see the **mmuserauth service create** command.

To change the authentication method that is already configured for object access, you need to remove the authentication method and ID mappings. For more information, see "Deleting the authentication and the ID mapping configuration" on page 165.

## Configuring AD without TLS for object access

Configuring AD without TLS does not provide secured communication between the IBM Spectrum Scale system and the authentication server.

1. Submit the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
--base-dn "dc=IBM,DC=local" --ks-dns-name c40bbc2xn3 --ks-admin-user admin
--servers myADserver --user-id-attrib cn --user-name-attrib sAMAccountName
--user-objectclass organizationalPerson --user-dn "cn=Users,dc=IBM,dc=local"
--ks-swift-user swift --ks-swift-pwd Passw0rd
```

   The system displays the following output:

```
Object configuration with LDAP (Active Directory) as identity
backend is completed successfully.
Object Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS               VALUES
-----------------------------------------------

OBJECT access configuration: AD
PARAMETERS               VALUES
-----------------------------------------------
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        false
ENABLE_KS_SSL            false
USER_NAME                cn=Administrator,cn=Users,dc=IBM,dc=local
SERVERS                  myADserver
BASE_DN                  dc=IBM,DC=local
USER_DN                  cn=users,dc=ibm,dc=local
USER_OBJECTCLASS         organizationalPerson
USER_NAME_ATTRIB         sAMAccountName
USER_ID_ATTRIB           cn
USER_MAIL_ATTRIB         mail
USER_FILTER              none
ENABLE_KS_CASIGNING      false
KS_ADMIN_USER            admin
```

## Configuring AD with TLS for object access

Configuring AD with TLS helps to encrypt the communication between the IBM Spectrum Scale system and AD server.

Configures AD with TLS as the authentication method for object access.

1. Ensure that the CA certificate for AD server is placed under `/var/mmfs/tmp` directory with the name `ldap_cacert.pem` specifically on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

   ```
   # stat /var/mmfs/tmp/ldap_cacert.pem
   File: ⌂/var/mmfs/tmp/ldap_cacert.pem⌂
   Size: 2130 Blocks: 8 IO Block: 4096 regular file
   Device: fd00h/64768d Inode: 103169903 Links: 1
   Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
   Context: unconfined_u:object_r:user_tmp_t:s0
   Access: 2015-01-23 12:37:34.088837381 +0530
   Modify: 2015-01-23 12:16:24.438837381 +0530
   Change: 2015-01-23 12:16:24.438837381 +0530
   ```

2. To configure AD with TLS authentication for object access, issue the **mmuserauth service create** command as shown in the following example:

   ```
   # mmuserauth service create --type ad --data-access-method object
   --user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
   --base-dn "dc=IBM,DC=local" --enable-server-tls --ks-dns-name c40bbc2xn3
   --ks-admin-user admin --servers myADserver --user-id-attrib cn
   --user-name-attrib sAMAccountName --user-objectclass organizationalPerson
   --user-dn "cn=Users,dc=IBM,dc=local" --ks-swift-user swift
   --ks-swift-pwd Passw0rd
   ```

   The system displays the following output:

   ```
   Object configuration with LDAP (Active Directory) as identity
   backend is completed successfully.
   Object Authentication configuration completed successfully.
   ```

3. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

   ```
   # mmuserauth service list
   ```

   The system displays the following output:

```
FILE access not configured
PARAMETERS              VALUES
-----------------------------------------------

OBJECT access configuration: AD
PARAMETERS              VALUES
-----------------------------------------------
ENABLE_ANONYMOUS_BIND   false
ENABLE_SERVER_TLS       true
ENABLE_KS_SSL           false
USER_NAME               cn=Administrator,cn=Users,dc=IBM,dc=local
SERVERS                 myADserver
BASE_DN                 dc=IBM,DC=local
USER_DN                 cn=users,dc=ibm,dc=local
USER_OBJECTCLASS        organizationalPerson
USER_NAME_ATTRIB        sAMAccountName
USER_ID_ATTRIB          cn
USER_MAIL_ATTRIB        mail
USER_FILTER             none
ENABLE_KS_CASIGNING     false
KS_ADMIN_USER           admin
```

## Configuring an LDAP-based authentication for object access

You can configure Keystone with an external LDAP server as the authentication back-end. This will allow LDAP users to access the object store using their LDAP credentials. The same LDAP server can be used for both object access and file access.

### Prerequisites

Ensure that you have the following details before you configure LDAP-based authentication:

- LDAP server details such as IP address or host name, LDAP user name, user password, base dn, and user dn.
- If you want to configure TLS with LDAP for secure communication between Keystone and LDAP, you need to place the CA certificate that is used for signing the LDAP server setup for TLS under the following directory of the node on which the **mmuserauth service create** command is run:
  - `/var/mmfs/tmp/ldap_cacert.pem`
- The secret key you provided for encrypting/decrypting passwords unless you have disabled prompting for the key.

See “Integrating with LDAP server” on page 130 for more information on the prerequisites for integrating LDAP server with the IBM Spectrum Scale system.

You need to issue the **mmuserauth service create** command to configure LDAP-based authentication with the following parameters:

- **--type** `ldap`
- **--data-access-method** `object`
- **--servers** `IP address or host name of LDAP` (all user lookups by Keystone is done only against this server. If multiple servers are specified, only the first server is used and rest are ignored).
- **--base-dn** `ldapBase`
- { **--enable-anonymous-bind** | **--user-name BindDN --password BindPwd**} (You need to mention either anonymous bind or either **--user-name** or **--password**).
- **--enable-server-tls**, if TLS needs to be enabled.
- **--user-dn** `ldapUserSuffix` (LDAP container from where users are looked up)
- **--ks-dns-name** `keystoneDNSName`
- **--ks-admin-user** `keystoneAdminUser` from LDAP.

- **--enable-ks-ssl**, if SSL needs to be enabled. You need to have another set of certificates that are placed in the standard directory.
- **--enable-ks-casigning**, if you want to use external CA signed certificate for token signing.
- **--ks-swift-user** swiftServiceUser from LDAP.
- **--ks-swift-pwd** swiftServiceUser Password from LDAP.

For more information on each parameter, see the **mmuserauth service create** command.

To change the authentication method that is already configured for object access, you need to remove the authentication method and ID mappings. For more information, see "Deleting the authentication and the ID mapping configuration" on page 165.

## Configuring LDAP without TLS for object access

Perform the following steps to configure LDAP-based authentication for object access:

1. To configure LDAP-based authentication for object access, issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=essldapdomain" --password "Passw0rd"
--base-dn dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com --ks-dns-name c40bbc2xn3
--ks-admin-user mamdouh --servers 192.0.2.11
--user-dn "ou=People,dc=essldapdomain" --ks-swift-user swift
--ks-swift-pwd Passw0rd
```

   The system displays the following output:

```
Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.
```

2. To verify the authentication configuration, issue the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

   The system displays the following output:

```
FILE access not configured
PARAMETERS               VALUES
-------------------------------------------------

OBJECT access configuration : LDAP
PARAMETERS               VALUES
-------------------------------------------------
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        false
ENABLE_KS_SSL            false
USER_NAME                cn=manager,dc=essldapdomain
SERVERS                  192.0.2.11
BASE_DN                  dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com
USER_DN                  ou=people,dc=essldapdomain
USER_OBJECTCLASS         posixAccount
USER_NAME_ATTRIB         cn
USER_ID_ATTRIB           uid
USER_MAIL_ATTRIB         mail
USER_FILTER              none
ENABLE_KS_CASIGNING      false
KS_ADMIN_USER            mamdouh
```

## Configuring LDAP with TLS for object access

Perform the following steps to configure LDAP with TLS-based authentication for object access:

1. Ensure that the CA certificate for the LDAP server is placed under /var/mmfs/tmp directory with the name ldap_cacert.pem specifically on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```
# stat /var/mmfs/tmp/ldap_cacert.pem
File: ⌂/var/mmfs/tmp/ldap_cacert.pem⌂
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. To configure LDAP with TLS-based authentication for object access, issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=essldapdomain" --password "Passw0rd"
--base-dn dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com --enable-server-tls
--ks-dns-name c40bbc2xn3 --ks-admin-user mamdouh --servers 192.0.2.11
--user-dn "ou=People,dc=essldapdomain" --ks-swift-user swift
--ks-swift-pwd Passw0rd
```

   The system displays the following output:

```
Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.
```

3. To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

   The system displays the following output:

```
FILE access not configured
PARAMETERS              VALUES
-------------------------------------------------

OBJECT access configuration : LDAP
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_ANONYMOUS_BIND   false
ENABLE_SERVER_TLS       true
ENABLE_KS_SSL           false
USER_NAME               cn=manager,dc=essldapdomain
SERVERS                 192.0.2.11
BASE_DN                 dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com
USER_DN                 ou=people,dc=essldapdomain
USER_OBJECTCLASS        posixAccount
USER_NAME_ATTRIB        cn
USER_ID_ATTRIB          uid
USER_MAIL_ATTRIB        mail
USER_FILTER             none
ENABLE_KS_CASIGNING     false
KS_ADMIN_USER           mamdouh
```

## Configuring object authentication with an external keystone server

The object protocol can be configured with an external keystone server. This can be accomplished by either using an existing internal keystone server that is already deployed in the local environment or by utilizing an external keystone server that is hosted outside of the local environment.

The following prerequisites must be met before you start configuring an external keystone server with the IBM Spectrum Scale system.

- The external keystone server must be running and reachable from all protocol nodes.

- The keystone server administrator must create an object storage service for the required user, for object authentication configuration.

To configure an external keystone server with the IBM Spectrum Scale system, issue the **mmuserauth service create** command as shown in the following example:

```
mmuserauth service create –data-access-method object –type
userdefined --ks-swift-user <SWIFTserviceUser> --ks-swift-pwd <SWIFTserviceUserpassword>
-ks-ext-endpoint <endpoint of keystone server>
```

### Configuring an external keystone server for object authentication when using the installation toolkit

If you plan to configure authentication for IBM Spectrum Scale for object storage with an external keystone server and you are using the installation toolkit, perform the following steps:

1. To configure an external keystone server for object authentication, start with a local authentication configuration by not providing the authentication object arguments to the IBM Spectrum Scale installation toolkit. `Local authentication` is the default for the installation toolkit.
2. Run the following commands to configure object authentication with external keystone:

```
mmuserauth service remove --data-access-method object

mmuserauth service remove --data-access-method object --idmapdelete

mmuserauth service create --data-access-method object --type userdefined
--ks-ext-endpoint http://specscaleswift.example.com:35357/v3
--ks-swift-user swift --ks-swift-pwd password
```

**Note:** Cleaning up authentication leads to loss of data access to the end clients. For example, in the preceding command sequence, client access to data created with local authentication enabled is lost when you remove local authentication before configuring external keystone.

### Configuring IBM Spectrum Scale for object storage with SSL-enabled external keystone

1. Remove the object authentication along with the ID mapping ID if it is present by running one of the following commands:

```
mmuserauth service remove --data-access-method object
```

```
mmuserauth service remove --data-access-method object --idmapdelete
```

2. Copy the CA certificate with the external keystone on the node where the **mmuserauth** command is being run.

   The location and the name of the CA certificate on the current node is `/var/mmfs/tmp/ks_ext_cacert.pem`.

3. Configure the object authentication by running the **mmuserauth service create** command with the `--enable-ks-ssl` option:

```
mmuserauth service create --data-access-method object --type userdefined
--ks-ext-endpoint https://specscaleswift.example.com:35357/v3
--ks-swift-user swift --ks-swift-pwd password --enable-ks-ssl
```

**Note:** Object configuration with SSL-enabled external keystone is not supported on the installer toolkit and **mmcesobjcrbase**.

## Creating object accounts

An account is used to group or isolate object resources. Each object user is part of an account. Object users are mapped to an account and can access only the objects that reside within the project. Each user needs to be defined with a set of user rights and privileges to perform a specific set of operations on the resources of the account to which it belongs. Users can be assigned to multiple accounts with different roles on each account.

You must create at least one account before adding users. An account contains a list of containers in the object storage. You can also define quota at the account level. An object account represents a storage location for a project rather than a specific user.

**Note:**
To work with this function in the IBM Spectrum Scale GUI, log on to the GUI and select **Object** > **Accounts**.

1. To view the details for an existing account, issue the **swift stat** command:

```
swift stat --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3 \
               --os-project-name admin \
               --os-project-domain-name Default \
               --os-username admin \
               --os-user-domain-name Default \
               --os-password Passw0rd \
               --auth-version 3
```

   or

```
source openrc
swift stat
```

   The system displays output similar to the following:

```
Account: AUTH_bea5a0c632e54eaf85e9150a16c443cet
```

   **Note:** To avoid specifying the option to the swift command, you can use the ~/openrc file from the protocol node. The swift command looks like:

```
source ~/openrc
swift stat
```

2. To create a new account, do the following steps:
   a. Use the **openstack project create** command to create a project.

      For example, create the project 'salesproject' in the Default domain using the command:

```
# openstack project create salesproject --domain Default
```

      The system displays output similar to the following:

```
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | ec4a0bff137b4c1fb67c6fe8fbb6a37b |
| name        | salesproject                     |
+-------------+----------------------------------+
```

   b. Use the **openstack role add** command to associate roles to the users who need access to the project:

```
 # openstack role add --user admin --project salesproject admin
```

      The system displays output similar to the following:

```
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description |                                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | ec4a0bff137b4c1fb67c6fe8fbb6a37b |
| name        | salesproject                     |
+-------------+----------------------------------+
```

3. To see the new account details, issue the **swift stat** command with the new project value:

```
# swift stat --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3 \
               --os-project-name salesproject \
               --os-project-domain-name Default \
```

```
--os-username admin \
--os-user-domain-name Default \
--os-password Passw0rd \
--auth-version 3
```

The system displays output similar to the following:

```
Account: AUTH_ec4a0bff137b4c1fb67c6fe8fbb6a37b
```

# Managing object users, roles, and projects

IBM Spectrum Scale for object storage uses the keystone service for identity management. Keystone provides user authentication and authorization processes.

You can use an external Microsoft Active Directory or LDAP server or a local database as the back-end to store and manage user credentials for user authentication. The authorization details such as relation of users with projects and roles are maintained locally by the keystone server. The customer can select the authentication server to be used. For example, if AD is already configured in the environment and the users who need access to the object store are part of AD, then the customer can configure Keystone with AD as the authentication and authorization back-end.

When the back-end authentication server is AD or LDAP, the user management operations such as creating or deleting a user are the responsibility of the AD/LDAP administrator, who can optionally also be the Keystone server administrator. When local authentication is used for object access, the user management operations are done by the Keystone administrator. In case of authorization, the management tasks such as creating roles, projects, and associating the user with them is done by the Keystone Administrator. The Keystone administration can be done through the Keystone V3 REST API or by using an OpenStack python-based client.

Before you start creating object users, and projects, ensure that Keystone server is configured and the authentication servers are set up properly.

**Note:**

- If the cluster is reachable from the system, the OpenStack command can be issued from any system.
- If the OpenStack command is run from any of the protocol nodes, then you can use the `openrc` file to set the required environment that is used by OpenStack commands to manage the Keystone server. The advantage of using the `openrc` file is that you are not required to enter the following details every time you enter the commands: `--os-identity-api-version`, `--os-username`, `--os-password`, `--os-project-domain-name`, `--os-user-domain-name`, `--os-domain-id default`, and `--os-auth-url`.
- The user create, update, and delete operations are only applicable when local authentication method is used for object access.
- For more information on the Keystone V3 REST API, see the OpenStack API Documentation (developer.openstack.org/api-ref-identity-v3.html).

## Creating a new user

When creating a new user in the local database to support local authentication for object access, activate the openrc file located under `/root/openrc` by default. You can load the openrc profile by running: **'source /root/openrc'** and this will automatically load the required environmental variables into your current location. The results will look similar to this:

```
export OS_AUTH_URL="http://cesobjnode:35357/v3"
export OS_IDENTITY_API_VERSION=symph
export OS_AUTH_VERSION=3
export OS_USERNAME="admin"
export OS_PASSWORD="Passw0rd"
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_NAME=Default
```

Use the **openstack user create** command and manually enter the parameters as shown in the following example to create new user in the local database to support local authentication for object access.

```
# openstack --os-identity-api-version 3 --os-username admin --os-password
Passw0rd --os-project-domain-name Default --os-user-domain-name Default --osdomain-
id default --os-auth-url http://specscaleswift.example.com:35357/v3 user create --password-prompt
--email newuser1@localdomain.com --domain default newuser1
User Password:
Repeat User Password:
+-----------+-----------------------------------------------------------------------------+
| Field     | Value                                                                       |
+-----------+-----------------------------------------------------------------------------+
| domain_id | default                                                                     |
| email     | newuser1@localdomain.com                                                    |
| enabled   | True                                                                        |
| id        | 2a3ef8031359457292274bcd70e34d00                                            |
| links     | {u'self':
  u'http://specscaleswift.example.com:35357/v3/users/2a3ef8031359457292274bcd70e34d00'}    |
| name      | newuser1                                                                    |
+-----------+-----------------------------------------------------------------------------+
```

**GUI navigation**

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

## Listing users

Use the **openstack user list** command as shown in the following example to list users who are created in the local database:

```
# source $HOME/openrc
# openstack user list
+----------------------------------+----------+
| ID                               | Name     |
+----------------------------------+----------+
| 2a3ef8031359457292274bcd70e34d00 | newuser1 |
| a95783144edd414aa236a3d1582a3067 | admin    |
+----------------------------------+----------+
```

## Changing the password of a user

Use the **openstack user set** command to update the object user details. The following example shows how to change the password:

```
# openstack user set --password Passw0rd newuser2
```

## Deleting a user

Use the **openstack user delete** command as shown in the following example to delete the users who are created in the local database:

```
# openstack user delete newuser2
```

## Listing user roles

Use the **openstack role list** command as shown in the following example to list the user roles:

```
# openstack role list
+----------------------------------+-------+
| ID                               | Name  |
+----------------------------------+-------+
| ed38022b46094a51918e6e46f87e7290 | admin |
+----------------------------------+-------+
```

## Creating a new role

Perform the following steps to create a new user role:

1. Issue the **openstack role create** command to create a new user role:

```
#openstack role create member
+-------+-----------------------------------------------------------------+
| Field | Value                                                           |
+-------+-----------------------------------------------------------------+
| id    | 1f14f95826fe4c8590760b3d3e4ce7e0                                |
| links | {u'self': u'http://specscaleswift.example.com:35357/v3/roles/1f14f95826fe4c85907
60b3d3e4ce7e0'}                                                         |
| name  | member                                                          |
+-------+-----------------------------------------------------------------+
```

2. Verify the newly created role by using the **openstack role list** command:

```
# openstack role list
+----------------------------------+--------+
| ID                               | Name   |
+----------------------------------+--------+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member |
| ed38022b46094a51918e6e46f87e7290 | admin  |
```

**GUI navigation**

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

## Assigning a role to a user

Perform the following steps to assign a user role to a user:

1. Issue the **openstack role add** command to assign role to a user as shown in the following example:

```
# openstack role add --user newuser1
--domain default member
```

2. Submit the **openstack role list** command to verify the user role of the user as shown in the following example:

```
# openstack role list --user newuser1
+----------------------------------+--------+
| ID                               | Name   |
+----------------------------------+--------+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member |
+----------------------------------+--------+
```

## Creating a new project, adding a user, and assigning a role to the user

Perform the following steps to create a new project and add a user to the project with a specified role:

1. Submit the **openstack project create** command to create a new project:

```
# openstack project create newproject
+-------------+----------------------------------------------------------------------------+
| Field       | Value                                                                      |
+-------------+----------------------------------------------------------------------------+
| description |                                                                            |
| domain_id   | default                                                                    |
| enabled     | True                                                                       |
| id          | 2dfcbdb70b75435fb2015c86d46ffc0b                                           |
| links       | {u'self':
u'http://specscaleswift.example.com:35357/v3/projects/2dfcbdb70b75435fb2015c86d46ffc0b'} |
| name        | newproject                                                                 |
| parent_id   | None                                                                       |
+-------------+----------------------------------------------------------------------------+
```

2. Submit the **openstack role add** command to add a role to the user as shown in the following example:

```
# openstack role add --user newuser1 --
project newproject member

# openstack role add --user newuser1
--project newproject admin
```

3. Submit the **openstack role list** command to list the user roles as shown in the following example:

```
# openstack role list --user newuser1 --
project newproject
+----------------------------------+--------+------------+----------+
| ID                               | Name   | Project    | User     |
+----------------------------------+--------+------------+----------+
| 1f14f95826fe4c8590760b3d3e4ce7e0 | member | newproject | newuser1 |
| ed38022b46094a51918e6e46f87e7290 | admin  | newproject | newuser1 |
+----------------------------------+--------+------------+----------+
```

### Listing endpoints

Use the **openstack endpoint list** command as shown in the following example to view the endpoints that are available:

```
# openstack endpoint list
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------+
| ID         | Region | Service Name | Service Type | Enabled | Interface | URL                                                 |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------+
| c36e..9da5 | None   | keystone     | identity     | True    | public    | http://specscaleswift.example.com:5000              |
| f4d6..b040 | None   | keystone     | identity     | True    | internal  | http://specscaleswift.example.com:35357             |
| d390..0bf6 | None   | keystone     | identity     | True    | admin     | http://specscaleswift.example.com:35357             |
| 2e63..f023 | None   | swift        | object-store | True    | public    | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s |
| cd37..9597 | None   | swift        | object-store | True    | internal  | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s |
| a349..58ef | None   | swift        | object-store | True    | admin     | http://specscaleswift.example.com:8080              |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------+
```

## Deleting expired tokens

By default, the Keystone Identity Service stores expired tokens in the database indefinitely. While potentially useful for auditing in production environments, the accumulation of expired tokens considerably increases the database size and might affect the service performance.

Use cron as follows to configure a periodic task on one of the protocol nodes that purges expired tokens hourly or based on the load in your environment.

```
# (crontab -l -u keystone 2>&1 | grep -q token_flush) || \
  echo '@hourly /usr/bin/keystone-manage token_flush >/var/log/keystone/keystone-tokenflush.log 2>&1' \
  >> /var/spool/cron/keystone
```

## Deleting the authentication and the ID mapping configuration

Deleting the authentication and ID mapping configuration results in loss of access to data. Before you remove or edit ID mappings, determine how access to data is going to be maintained.

### Removing file authentication

**Note:** You are not allowed to delete both the authentication configuration and the ID mappings at the same time. You need to remove the authentication configuration first and then the ID maps. The system does not allow you to delete the ID maps without deleting the authentication configuration.

1. Issue the **mmuserauth service list** command to see the authentication method that is configured in the system:

```
# mmuserauth service list
FILE access configuration: LDAP
PARAMETERS VALUES
-------------------------------------------------
ENABLE_ANONYMOUS_BIND false
ENABLE_SERVER_TLS false
ENABLE_KERBEROS false
USER_NAME cn=manager,dc=example,dc=com
```

```
SERVERS 10.0.100.121
NETBIOS_NAME eslhnode
BASE_DN dc=example,dc=com
USER_DN ou=people,dc=example,dc=com
GROUP_DN none
NETGROUP_DN ou=netgroup,dc=example,dc=com
USER_OBJECTCLASS inetOrgPerson
GROUP_OBJECTCLASS posixGroup
USER_NAME_ATTRIB cn
USER_ID_ATTRIB uid
KERBEROS_SERVER none
KERBEROS_REALM none
OBJECT access not configured
PARAMETERS VALUES
-------------------------------------------------
```

2. Issue the **mmuserauth service remove** command to remove the authentication configuration as shown in the following example:

```
# mmuserauth service remove -–data-access-method file
mmcesuserauth service remove: Command successfully completed.
```

3. Issue the **mmuserauth service list** command to verify whether the authentication configuration is removed:

```
# mmuserauth service list
FILE access not configured
PARAMETERS VALUES
-------------------------------------------------
OBJECT access not configured
PARAMETERS VALUES
-------------------------------------------------
```

For more information, see *mmuserauth command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Deleting authentication configuration as shown in the previous example does not delete the ID maps. Use the **--idmapdelete** option with the **mmuserauth service remove** command to remove ID maps that are created for user authentication:

```
# mmuserauth service remove --data-access-method file --idmapdelete
mmuserauth service remove: Command successfully completed
```

## Removing object authentication

The deletion of ID maps that are used for file access is only applicable when AD with Automatic ID mapping or RFC2307 ID mapping is configured.

Deleting ID maps might also be required in the case of object access. ID map delete option can be used if the system administrator wants to clean up the entire Keystone authentication configuration, including the mapping of users with projects and roles. Cleaning up of ID mapping information results in loss of access to any existing data that is being accessed through the Object Storage interface. Deleting ID mappings deletes user-role-projects mappings as well. Without these mappings, new users are unable to access the old data unless the keystone administrator creates the mapping again for the new user. ID maps are deleted in environments where the object protocol needs to be removed or the entire object store needs to be erased. This is usually done in preproduction or test environments.

If you want to change the authentication method that is already configured for object access, you must remove the authentication method and ID mappings by issuing the **mmuserauth service remove --data-access-method object** and **mmuserauth service remove --data-access-method object --idmapdelete** commands in sequence, as shown in the following example:

```
# mmuserauth service remove --data-access-method object
mmuserauth service remove: Command successfully completed
```

```
# mmuserauth service remove --data-access-method object --idmapdelete
mmuserauth service remove: Command successfully completed

# mmuserauth service list
FILE access not configured
PARAMETERS VALUES
-----------------------------------------------
OBJECT access not configured
PARAMETERS VALUES
-----------------------------------------------
```

**Note:** When you delete the ID maps that are created for file or object access, ensure that all the protocol nodes are in the healthy state. You can view the health status of protocol nodes by using the **mmces state show -a** command.

## Listing the authentication configuration

Use the **mmuserauth service list** command to see the authentication method that is configured in the system.

```
# mmuserauth service list
FILE access configuration : LDAP
PARAMETERS VALUES
-----------------------------------------------
ENABLE_SERVER_TLS false
ENABLE_KERBEROS false
USER_NAME cn=manager,dc=example,dc=com
SERVERS 9.122.123.172
NETBIOS_NAME eslhnode
BASE_DN dc=example,dc=com
USER_DN ou=people,dc=example,dc=com
GROUP_DN none
NETGROUP_DN ou=netgroup,dc=example,dc=com
USER_OBJECTCLASS inetOrgPerson
GROUP_OBJECTCLASS posixGroup
USER_NAME_ATTRIB cn
USER_ID_ATTRIB uid
KERBEROS_SERVER none
KERBEROS_REALM none
OBJECT access not configured
PARAMETERS VALUES
-----------------------------------------------
```

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

## Verifying the authentication services configured in the system

Use the **mmuserauth service check** command to check whether the authentication configuration is consistent across the cluster and the required services are enabled and running. This command validates and corrects the authentication configuration files and starts any associated services if needed.

You can check the following authentication details by using the **mmuserauth service check** command:

- `--data-access-method {file|object|all}` Authentication method.
- `[-N|--nodes] {node-list|cesNodes}` Authentication configuration on each node. If the specified node is not a protocol node, the check operation gets ignored on that node. If a 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.
- `--server-reachability` Verify whether the authentication backend server is reachable. If object is configured with external Keystone server, this check is not performed.

- `-[ r | --rectify ]` Rectify the configuration for the specified nodes by copying any missing configuration files or SSL/TLS certificates from another node.

For more information, see the topic *mmuserauth command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

**Example - File authentication check**

Issue the **mmuserauth service check** command.

```
# mmuserauth service check --type file --nodes dgnode3,dgnode2
--server-reachability -r

Userauth file check on node: dgnode3
Checking SSSD_CONF: OK
LDAP server status
LDAP server 9.118.46.17 : OK
Service 'sssd' status: OK
Userauth file check on node: dgnode2
dgnode2: not CES node. Ignoring...
```

You can use the **id** command to see the list of users and groups fetched from the LDAP server. For example:

```
# id ldapuser2
uid=1001(ldapuser2) gid=1001(ldapuser2) groups=1001(ldapuser2)
```

**Example - Object authentication check**

Issue the **mmuserauth service check** command.

```
# mmuserauth service check
--server-reachability --data-access-method
object
Userauth object check on node: dgnode3
Checking keystone.conf: OK
LDAP servers status
LDAP server sonash1 : OK
Service 'keystone-all' status: OK
```

## Modifying the authentication method

If data already exists or is created with the existing authentication and ID mapping method, it is not recommended to change the authentication or the ID mapping modes. Changing the authentication method also might invalidate the existing ACLs that are applicable to files and directories. ACLs depend on the preexisting users and group IDs.

To modify the authentication method, perform the following steps:

1. List the existing authentication configuration for file and object authentication method by using the **mmuserauth service list** command.
2. Identify the parameters that you need to change. If an authentication method and ID maps are already existing, you must not plan to change the authentication type or ID mapping schemes. When you remove the existing authentication method and ID maps, the user and group of users who were accessing the data cannot access the data anymore.

   The following list provides the parameters that can be modified in each authentication configuration.

   **For file authentication:**

   - With LDAP authentication, all attributes of the configuration can be modified. When changing authentication servers, ensure that the newly specified servers are the replica of the original servers, otherwise, it might result in loss of access to data.

- With AD authentication, all attributes of the configuration can be modified. When changing the authentication server, ensure that the newly specified server is a domain controller in the same AD domain that is being served by the original server, otherwise, it might result in loss of access to data. If UNIX ID maps are specified in current configuration and more new AD domains are to be added, it is vital to specify the current list of domains along with the new domains.
- With NIS authentication, all attributes of the configuration can be modified. When changing servers, ensure that the newly specified servers are serving the same NIS domain as the original servers; otherwise, it might result in loss of access to data.

**For object authentication:**

You can change all options except **--data-access-method** and **--type** parameters.

3. Clean up the existing authentication by using the **mmuserauth service remove** command. Do not specify the **--idmapdelete** option as it results in loss of access to data.
4. Issue the **mmuserauth service create** with the required parameter change; ensuring that you use the same authentication, ID mapping scheme, and associated authentication servers.
5. List the authentication configuration by using the **mmuserauth service list** to verify the change.
6. Ensure that the authentication is consistent across the cluster by using the **mmuserauth service check** command.

# Authentication limitations

Consider the following authentication limitations when you configure and manage the IBM Spectrum Scale system:

## Object access limitations

For Active Directory (AD)-based authentication for object access:

- Only single AD server is used. If the configured AD server is down, the Keystone authentication fails.
- Does not support multiple AD Domains.
- Only Windows 2008 R2 and later are supported.
- Only read access to the AD server is supported. That is, you are not authorized to create a new user and modify or delete an existing user from the IBM Spectrum Scale system. Only the AD server administrator can do these tasks.

For Lightweight Directory Access Protocol (LDAP)-based authentication for object access:

- Only single LDAP server is used. If the configured LDAP server is down, the Keystone authentication fails.
- Only LDAP servers compatible with LDAP RFC 4511 are supported.
- Only read access to the LDAP server is supported. That is, you are not authorized to create a new user and modify or delete an existing user from the IBM Spectrum Scale system. Only the LDAP server administrator can do these tasks.

## File access limitations

### AD based authentication

For AD with automatic ID mapping:

- No support for migrating the internally generated user and group ID maps to external ID mapping server. If data is stored on the IBM Spectrum Scale system with AD and automatic ID mapping, adding RFC2307 later requires the UIDs and GIDs that are used internally by the IBM Spectrum Scale system match the UIDs and GIDs stored in RFC2307. This is not possible if conflicting UIDs and GIDs are already stored in RFC2307. To avoid potential conflicts, configure the IBM Spectrum Scale system by using AD and RFC2307 right from the beginning.

- Although AD along with automatic ID mapping can be used to have same ID maps between systems that are in AFM relationship, this configuration does not serve as a complete replacement of RFC2307. This configuration can be used in a predominantly SMB only setup, where NFS users are not already present in the environment. If NFS users are preexisting in the customer environment and these users intend to access the data with SMB users, then RFC2307 is mandatory.
- When AD-based authentication is used, SMB protocol access is kerberized by default. Access the system by using the netbios name that is specified in the command.
- Kerberized NFSv3-based access is not supported with AD as an authentication server.

For AD with RFC2307:

- Enabling RFC2307 for a trusted domain requires a two-way trust between the native and the trusted domains.
- To access the IBM Spectrum Scale system, users and groups must have a valid UID/GID assigned to them in AD. For user access, the windows group membership are evaluated on the IBM Spectrum Scale system. Hence, accessing a user's primary group is considered as the Microsoft Windows Primary group and not the UNIX primary group that is listed in the UNIX attribute tab in the user's properties. Therefore, the user's primary Microsoft Windows group must be assigned with a valid GID.
- The **mmuserauth service create** command does not check the two-way trust between the native domain and the RFC2307 domain that is required for ID mapping services to function properly. The customer is responsible for configuring the two-way trust relationship between these domains. The customer is responsible for assigning UIDs to users and GIDs to groups. The command does not return an error if a UID or GID is not assigned.
- Multiprotocol access of protocol exports is only allowed between NFSV4 and SMB. That is, you cannot access the same export by using both NFSV3 and SMB protocols.
- Kerberized NFSv3-based access is not supported by AD as an authentication server.

**LDAP based authentication**

- Users with the same user name from different organizational units under the specified baseDN in the LDAP server are denied access to SMB shares irrespective of the LDAP user suffix and LDAP group suffix values configured on the system.
- If multiple LDAP servers are specified during configuration, at any point in time, only one LDAP server is used.
- LDAP referrals are not supported.
- ACL management through windows clients is not supported.
- Only LDAP servers that implement RFC2307 schema are supported.

**General limitations for file access**

- When the SMB service is stopped on a protocol node, with AD with RFC2307 authentication as the authentication method, the NFS based access also gets affected on that protocol node.
- When using Microsoft Active Directory (AD) as an authentication system, the IBM Spectrum Scale system supports only the NetBIOS logon name for authentication and not the User Principle Name (UPN). Active Directory replaces some of the special characters that are used in the UPN with the underscore character (hexadecimal value 0x5F) for the related NetBIOS logon name of the user. For the complete list of the special characters that are replaced in the NetBIOS logon name, see Microsoft Active Directory documentation. Follow these steps to locate the NetBIOS logon name for an Active Directory domain user:
  1. From the Windows Start menu, select Administrative Tools > Active Directory Users and Computers
  2. Right-click the Active Directory Domain user for which you require the NetBIOS logon name
  3. Select Properties > Account Tab and check the value of the User logon name (pre-Windows 2000): field

- Authentication configuration commands restart the IBM Spectrum Scale protocol services such as SMB and NFS. The protocol services resume a few seconds after an authentication configuration command completes.
- For file data access, switching or migrating from one authentication method to another is not supported, since it may lead to loss of access to the data on the system.
- The IBM Spectrum Scale system does not support authentication servers (AD, LDAP, and NIS) that are running on virtual machines that are stored on an SMB or NFS export. The IBM Spectrum Scale system requires the authentication server to be running, while configuring authentication and while serving connection requests over protocols. The virtualizer cannot boot the authentication server unless the protocols are configured for authentication and data is ready to be served over the exports.
- The length of a user name or a group name of the users and group of users who need to access the data cannot be more than 32 characters
- The NFSV4 clients must be configured with the same authentication and ID mapping server as that of the IBM Spectrum Scale system. The IBM Spectrum Scale system does not support an NFSV4 client configured with different authentication and ID mapping servers.
- Based on the hardware platform the protocol nodes are configured on, consider the group ID resolution as per the limitation described in the IBM Spectrum Scale FAQ. For more information, see IBM Spectrum Scale FAQs.
- In order to use NFSV4 ID mapping, the NFS ID map domain needs to be set on the IBM Spectrum Scale protocol nodes and the same NFS ID map domain must be configured on every NFS client. Below is an example of how to configure NFSV4 ID mapping.
  1. Issue the **mmnfs configuration list** command.

     The system will display this output showing that the ID map domain is not set:

     ```
     IDMAPD Configuration
     ====================
     ====================
     ```

  2. Enter the following command to set the NFS ID map domain:

     ```
     mmnfs configuration change IDMAPD_DOMAIN=MY_IDMAP_DOMAIN
     ```

  3. Issue the **mmnfs configuration list** command to verify that the ID map domain is set.

     The system displays this output:

     ```
     IDMAPD Configuration
     ========================
     DOMAIN: MY_IDMAP_DOMAIN
     ========================
     ```

# Chapter 15. Managing protocol data exports

You can manage the data exports that you have created using NFS, SMB, and Object.

## Managing SMB shares

All SMB administration commands can be run from any cluster node including non-CES nodes. However, the latency of the administration command execution on a CES node is lower as the administrative changes can be applied straight away. Use the following information to manage SMB shares in IBM Spectrum Scale.

**GUI navigation**

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **SMB Shares**.

## Creating SMB share

Use the following information to create an SMB share:

1. Create the directory to be exported through SMB:

   ```
   mmcrfileset fs01 fileset --inode-space=new
   mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
   mkdir /gpfs/fs01/fileset/smb
   ```

   **Note:** IBM recommends an independent fileset for SMB shares.

   Create a new independent fileset with these commands:

   ```
   mmcrfileset fs01 fileset --inode-space=new
   mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
   ```

   If the directory to be exported does not exist, create the directory first by running the following command:

   ```
   mkdir /gpfs/fs01/fileset/smb"
   ```

2. The recommended approach for managing access to the SMB share is to manage the ACLs from a Windows client machine. To change the ACLs from a Windows client, change the owner of the share folder to a user ID that will be used to make the ACL changes by running the following command:

   ```
   chown 'DOMAIN\smbadmin' /gpfs/fs01/fileset/smb
   ```

3. Create the actual SMB share on the existing directory:

   ```
   mmsmb export add smbexport /gpfs/fs01/fileset/smb
   ```

   Additional options can be set during share creation. For a list of all the supported SMB options, see *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*.

4. Verify that the share has been created:

   ```
   mmsmb export list
   ```

5. Access the share from a Windows client using the user ID that has been previously made the owner of the folder.
6. Right-click the folder in the Windows Explorer, open the **Security** tab, click **Advanced**, and modify the Access Control List as required.

   **Note:** An SMB share can only be created when the ACL setting of the underlying file system is **-k nfsv4**. In all other cases, **mmsmb export create** will fail with an error.

   See “Authorizing protocol users” on page 258 for details and limitations.

**GUI navigation**

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **SMB Shares**.

## Changing SMB share configuration

Use the following information to change the SMB share configurations.

For the documentation of all supported options, see *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*.

To see a list of supported configuration options for SMB shares, run the command:

```
mmsmb export change --key-info supported
```

For example, to change the descriptive comment for a share, run the command:

```
mmsmb export change smbshare --option 'comment=Project X export'
```

To list the configuration of all SMB shares, run the command:

```
mmsmb export list --all
```

**Note:** Changes to SMB share configurations only apply to client connections that have been established after the change has been made.

### GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **SMB Shares**.

## Creating SMB share ACLs

The SMB protocol supports a separate level of ACLs that can be optionally added to an SMB share.

For more information, see Managing ACLs of SMB exports using MMC.

For details, see the information about managing the SMB share ACLs from a Windows client through the MMC.

## Removing SMB shares

To remove an SMB share, use the **mmsmb** command. Use the following information to remove SMB shares.

To remove an SMB share:

1. Run the following command:
   ```
   mmsmb export remove smbexport
   ```
2. Verify that the export has been removed by listing the configured SMB share again:
   ```
   mmsmb export list
   ```

### GUI navigation

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **SMB Shares**.

## Listing SMB shares

To list the SMB shares, run the following command:

```
mmsmb export list
```

## Managing SMB shares using MMC

Microsoft Management Console (MMC) is a Windows tool that can be used to do basic configuration tasks on an SMB server. These tasks include administrative tasks such as listing or closing the connected users and open files, and creating and manipulating SMB shares. You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for managing SMB shares on the IBM Spectrum Scale cluster.

**Attention:** Listing a large number of entities (thousands of files, connections, locks, etc.) using Microsoft Management Console (MMC) might take a very long time and it might impact the performance of the file server. In these cases, it is recommended to use server-side administration tools.

Ensure that the following tasks are complete before you manage SMB shares:

- IBM Spectrum Scale is installed and configured.
- The SMB protocol is enabled and healthy SMB services are running on all protocol nodes.
- Required SMB shares are created and mounted from the Windows client.
- Microsoft Active Directory (AD) based authentication is set up. This includes:
  - Cluster nodes and client are domain members.
  - The client on which Microsoft Management Console (MMC) is running is a domain member.
  - Accurate DNS information is configured. If active sessions are listed, MMC tries to do a reverse pointer record lookup with DNS for every session (client IP), and if that fails then MMC hangs.
  - Involved NetBIOS names can be resolved using DNS.

  For using the Shared Folders Microsoft Management Console (MMC) snap-in, you must be a member of the local administrators group of the cluster. After joining the cluster to an AD domain, only the domain admins group is a member of the administrators group of the cluster.

  To add other users who can use the Shared Folders Microsoft Management Console (MMC) snap-in:

  1. Connect to MMC as a user that is a member of the domain admins group.
  2. Navigate to **System Tools** > **Local Users and Groups** and add a user to the local administrators group.

  For more information, see the Microsoft Management Console documentation.

The following MMC features are not supported for managing SMB shares on the IBM Spectrum Scale cluster:

- Audit of MMC read operations
- Event viewer
- Setting max connections per share

### Connecting to SMB shares by using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for connecting to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.

b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.

3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

   **Note:** If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see "Managing SMB shares using MMC" on page 175.

## Creating SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for creating SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the server on which you want to create SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, right-click **Shares** and then click **New Share**. The Create A Shared Folder wizard opens.

   **Note:** If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see "Managing SMB shares using MMC" on page 175.
4. In the Create A Shared Folder wizard, click **Next**.
5. In the **Folder path** field, enter the share path and click **Next**.

   **Note:** The directory for the SMB has to already exist in the file system.
6. Enter the SMB share name and description, select the required offline setting, and then click **Next**.
7. Select the required SMB share permission setting and click **Finish**.

## Modifying or removing SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for modifying or removing SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the server on which you want to create SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

   **Note:** If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see "Managing SMB shares using MMC" on page 175.
4. Do one of the following steps depending on whether you want to modify or remove SMB shares:
   - To modify an SMB share:

a. In the right pane, right-click the SMB share that you want to modify, and then click **Properties**.
b. Modify the properties as required and click **OK**.

- To remove an SMB share:
  a. In the right pane, right-click the SMB share that you want to remove, and then click **Stop Sharing**.

## Managing ACLs of SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for managing access control lists (ACLs) of SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

   **Note:** If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see "Managing SMB shares using MMC" on page 175.
4. In the right pane, right-click the SMB share for which you want to view or change the permissions and then click **Properties**.
5. You can do one of the following:
   - To view the permissions a user or a group has for the SMB share, on the **Share Permissions** tab, under the "Group or user names" pane, click on the user name or the group name.

     The permissions are displayed in the "Permissions for" pane.
   - To change the permissions a user or a group has for the SMB share, on the **Security** tab, under the "Group or user names" pane, click on the user name or the group name and then click **Edit**.

     **Note:** Changes affect only the SMB share, not the ACL in the file system of the exported directory.

     For information on permissions that you can change, see documentation for the Shared Folders Microsoft Management Console (MMC) snap-in.

## Modifying offline settings of SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for modifying offline settings of SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Shares**. All SMB shares are listed in the right pane.

**Note:** If there is a permissions related error when you click **Shares**, verify that you are a member of the local administrators group of the cluster. For more information, see "Managing SMB shares using MMC" on page 175.

4. In the right pane, right-click the SMB share whose offline settings you want to modify, and then click **Properties**.
5. On the **General** tab, click **Offline Settings**.
6. In the Offline Settings window, configure the offline settings of the SMB share. For information on offline settings that you can configure, see documentation for the Shared Folders Microsoft Management Console (MMC) snap-in.

## Viewing active connections to SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing active connections to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Sessions**. All active connections to SMB shares are listed in the right pane.

## Disconnecting active connections to SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for disconnecting active connections to SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Sessions**. All active connections to SMB shares are listed in the right pane.
4. In the right pane, right-click the connection that you want to close and then click **Close Session**.

   **Attention:** If connections are forced to close, data loss might occur for open files on the connections getting closed.
5. Click **OK** to confirm.

## Viewing open files in SMB shares using MMC

You can use Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing open files in SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:

a. Click **Action** > **Connect to another computer**.
b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.

3. In the left pane, click **Open Files**. All open files in SMB shares are listed in the right pane.

### Viewing the number of locks on files in SMB shares using MMC

You can use the Shared Folders Microsoft Management Console (MMC) snap-in on Microsoft Windows clients for viewing the number of locks on open files in SMB shares on the IBM Spectrum Scale cluster.

1. Open the Shared Folders Microsoft Management Console (MMC) snap-in:
   a. Click **Start** > **Run**.
   b. Type `fsmgmt.msc` and click **OK**.

   The Shared Folders Microsoft Management Console (MMC) snap-in opens.
2. Connect to the IBM Spectrum Scale cluster that has the SMB shares:
   a. Click **Action** > **Connect to another computer**.
   b. Type the IP address of the server you want to connect to in the **Another computer** field and click **OK**.
3. In the left pane, click **Open Files**. All open files in SMB shares are listed in the right pane.
4. In the right pane, view locks on a file under the **# Locks** column.

   The number of locks is displayed under the **# Locks** column and the type of locks is displayed under the **Open Mode** column.

## SMB share limitations

When you create SMB shares, consider their limitations and support restrictions.

- NTFS alternate data streams are not supported. For example, named streams generated by a Mac OS X operating system cannot be stored directly.
- The encryption status of files cannot be queried or changed from SMB clients. Use the **mm CLI** commands instead.
- When propagation of opportunistic locks across protocols is enabled (SMB option gpfs:leases), then Level 2 oplocks are not granted and Exclusive or batch oplocks are not broken down to Level 2 oplocks and are revoked from the system.
- Symbolic links cannot be created or changed from SMB clients and are not reported as symbolic links.
- Symbolic links created via NFS or directly in the file system will be respected as long as they point to a target under the same shared directory.
- Distributed File System (DFS) is not supported.
- Windows Internet Name Service (WINS) is not supported.
- Retrieving Quota information using NT_TRANSACT_QUERY_QUOTA is not supported.
- Setting Quota information using NT_TRANSACT_SET_QUOTA is not supported.
- Setting the maximum number of connections to a share is not supported. The MMC GUI allows specifying this parameter, but it cannot be set on an IBM Spectrum Scale cluster.
- Unix Extensions are not supported.
- You cannot create a shadow copy of a shared folder using a remote procedure call from a shadow copy client. Backup utilities, such as Microsoft Volume Shadow Copy Service, cannot create a shadow copy of a shared folder using a procedure call.
- The Branch Cache hash operations using SRV_READ_HASH IOCTL are not supported.
- Leases are not supported.
- Only the SMB2 and SMB3 protocol versions are supported.
- Only mandatory SMB3 protocol features are supported.
- No support of dynamic ACLs and SACLs.

- No support of SID history.
- No support of SMB 3.1.

SMB Clients:
- Windows: SMB 1 is not supported, thus no access for Windows XP clients
- Linux: SMB1 is not supported, so make sure to use the version option with the kernel SMB client:
  ```
  mount.cifs //fscc-p8-11/ralph /media/ss -o user=aduser1,pass=Passw0rd,dom=W2K8DOM05,vers=2.0
  ```

## Managing NFS exports

Use the following information to manage NFS exports in IBM Spectrum Scale.

## Creating NFS exports

To add an NFS export, use the **mmnfs** export add command.

1. If the directory to be exported does not exist, create the directory by running the following commands:
   ```
   mmcrfileset fs01 fileset --inode-space=new
   mmlinkfileset fs01 fileset -J /gpfs/fs01/fileset
   ```
   For more details, see *mmcrfileset command* and *mmlinkfileset command* in *IBM Spectrum Scale: Command and Programming Reference*.

   **Note:** We recommend an independent fileset for NFS exports.
2. Adjust the ownership and permissions of the folder as required.
   Use the GPFS ACL's with **mmgetacl** and **mmputacl** to set the correct ownership and the access permission.
   Additional options can be set during export creation. For the documentation of all supported options, see *mmnfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.
3. Create the NFS export using the following command:
   ```
   mmnfs export add /gpfs/fs01/fileset
   ```

**GUI navigation**

To work with this function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **NFS Exports**.

## Changing NFS export configuration

After an NFS export is created, the export attributes can be changed by using the **mmnfs export change** command.

For the documentation of all supported options, see *mmnfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

For example, to grant another client IP address access to the NFS export, run the following command:
```
mmnfs export change /gpfs/fs01/nfs --nfsadd 10.23.23.23
```

After the change is made, verify the configuration by running the following command:
```
mmnfs export list
```

The system displays output similar to this:

```
Path Delegations Clients
------------------------------
/gpfs/fs01/nfs none 10.23.23.21
/gpfs/fs01/nfs none 10.23.23.22
/gpfs/fs01/nfs none 10.23.23.23
```

Changing the NFS export configuration is not dynamic; NFS services automatically restart during the **mmnfs export change** command. This means that an NFS client that did a soft mount might lose connectivity, which might result in application failures; an NFS client that did a hard mount might "stall" during the NFS grace period.

The **mmnfs export change** command currently does not update the export in a dynamic fashion on the running instance of NFS; it stops and starts NFS on all CES nodes on which the NFS server is currently running.

## Removing NFS exports

To remove an NFS export, use the **mmnfs export remove** command.

To remove an NFS export, follow these steps:

1. Enter the following command:
   ```
   mmnfs export remove /gpfs/fs01/nfs
   ```
   The system displays output similar to the following:
   ```
   The NFS export was deleted successfully.
   ```
2. Verify that the export has been removed by listing the configured NFS exports:
   ```
   mmnfs export list
   ```

## Listing NFS exports

To list the NFS exports, enter the following command:

```
mmnfs export list
```

The system displays output similar to the following:

```
Path           Delegations Clients
---------------------------------------
/ibm/fs1/fset1 none        10.0.0.1
/ibm/fs1/fset1 none        10.0.0.2
/ibm/fs1/fset1 none        *
```

## GUI navigation for NFS exports

Use the following information to manage NFS exports in IBM Spectrum Scale.

To work with the NFS exports function in the GUI, log on to the IBM Spectrum Scale GUI and select **Protocols** > **NFS Exports**.

---

## Multiprotocol exports

Exports for SMB and NFS protocols can be configured so that they have access to the same data in the GPFS file system.

To export data via multiple protocols, first create an export for one protocol using the appropriate GPFS command (for example, **mmnfs export add**). In order to export the same GPFS path via a second protocol, simply create another export using the protocol-specific export management command (for example, **mmsmb export add**).

The operations of adding and removing exports do not delete any data in the GPFS file system. If at a later time access to a GPFS file system for a specific protocol needs to be removed, this can be done via the corresponding command. Removal of exports is a logical operation and does not change the data in the GPFS file system. It also does not impact access to the same data configured via another protocol.

## Multiprotocol export considerations

Exports for SMB and NFS protocols can be configured so that they have access to the same data in the file system. In addition, the data can be accessed directly in the file system on the cluster nodes. When configuring access to the same GPFS file system via both the NFS and SMB protocols, certain limitations apply.

These restrictions apply to the general areas of file locking (including share reservation and lock semantics), recovery (reclaim), and cross-protocol notifications.

SMB in IBM Spectrum Scale can be configured to maintain locks in SMB rather than GPFS. In this mode NFS is not aware of SMB locks and therefore it should not be used with concurrent access via NFS.

Furthermore, SMB is not aware of NFS grace periods (in which NFS clients are given time to reclaim any locks and share reservations). If you expect a lot of contention between SMB and NFS, NFSv4 reclaims might fail.

The NFS server always relies on the GPFS file system for managing file locks and uses the GPFS internal mechanism to synchronize these locks across all the nodes in the cluster. The NFS server is informed by GPFS of any changes on the file system objects, so NFS clients when obtaining file system information always are presented with the latest status.

**Note:** The SMB shares must be configured with gpfs:leases=yes and gpfs:sharemodes=yes. This implies that no level 2 oplocks will be granted to SMB clients.

# Chapter 16. Managing object storage

Use the following information to use and manage the IBM Spectrum Scale for object storage features.

## Understanding and managing Object services

Use the following information to manage services related to IBM Spectrum Scale for object storage.

IBM Spectrum Scale uses the **mmces service** command to enable, start, stop, or disable Object services on all protocol nodes.

The enable and disable operations are cluster-wide operations. To enable or disable the Object protocol, use **mmces service [enable | disable] OBJ**. The Object protocol must have been initially configured using the **mmobj swift base** command before it can be enabled in the cluster.

**CAUTION:**
**Disabling the object service unconfigures the Object protocol and discards OpenStack Swift configuration and ring files from the CES cluster. If Openstack Keystone configuration is configured locally, disabling object storage also discards the Keystone configuration and database files from the CES cluster. However, to avoid accidental data loss, the associated filesets used for the object data are not automatically removed during disable. The filesets for the object data and any filesets created for optional object storage policies need to be removed manually. For enabling the object service subsequently, either different fileset names need to be specified or the existing filesets need to be cleaned up. For information on cleaning up the object filesets, see the steps** *"Remove the fileset created for object"* **and** *"Remove any fileset created for an object storage policy"***(if applicable) in the** *Cleanup procedures required if reinstalling with the spectrumscale installation toolkit* **topic of** *IBM Spectrum Scale: Concepts, Planning, and Installation Guide***.**

**Note:** To disable the object protocol, first remove the object authentication. For complete usage information, see the *mmuserauth command* in the *IBM Spectrum Scale: Command and Programming Reference*.

In addition, enabled Object service can be started and stopped on individual nodes or cluster-wide.

To start or stop the Object protocol cluster-wide. use -a flag **mmces service [start | stop] OBJ -a**.

To start or stop the Object protocol on individual nodes, use **mmces service [start | stop] OBJ -N <node>**.

**Attention:** If object services on a protocol node are stopped by the administrator manually, access to object data might be impacted unless the CES IP addresses are first moved to another node. There are multiple ways to accomplish this, but the simplest is to suspend the node. After suspending a node, CES automatically moves the CES IPs to the remaining nodes in the cluster. However, doing this suspends operation for all protocols running on that protocol node.

If you want to stop object services on a protocol node, you can use the following steps:

1. Suspend CES operations on the protocol node using the **mmces node suspend** command.
2. View the CES IP addresses on that node using the **mmces address list** command and verify that all CES IP addresses have been moved to other protocol nodes.
3. Stop the object services using the **mmces service stop OBJ** command.

**Note:** All object services must be controlled by using only the mmces service start/stop and systemctl commands. The use of swift-init command is not supported and might cause your system to operate

incorrectly.
Performing these steps ensures that object functionality is available on other nodes in the cluster.

To restore object services on that protocol node, you can use the following steps:

1. Resume CES operations on the protocol node using the **mmces node resume** command.
2. View the CES IP addresses on that node using the **mmces address list** command and verify that all CES IP addresses have been moved to that protocol node.
3. Start the object services using the **mmces service start OBJ** command.

Use the **mmces service list** command to list the protocols enabled on IBM Spectrum Scale. List a verbose output of object services running on the local node using the **-v** flag as shown in the following example:

```
# mmces service list -v
Enabled services: OBJ SMB NFS
 OBJ is running
 OBJ:openstack-swift-object                  is running
 OBJ:openstack-swift-account                 is running
 OBJ:openstack-swift-container               is running
 OBJ:openstack-swift-proxy                   is running
 OBJ:memcached                               is running
 OBJ:openstack-swift-object-replicator       is running
 OBJ:openstack-swift-account-reaper          is running
 OBJ:openstack-swift-account-replicator      is running
 OBJ:openstack-swift-container-replicator    is running
 OBJ:openstack-swift-object-sof              is running
 OBJ:httpd (keystone)                        is running
 SMB is running
 NFS is running
```

For complete usage information, see *mmces command* in *IBM Spectrum Scale: Command and Programming Reference*.

Every object protocol node can access every virtual device in the shared file system, and some OpenStack Swift object services can be optimized to take advantage of this by running from a single Object protocol node.

Even though objects are not replicated by OpenStack Swift, the **swift-object-replicator** runs to periodically clean up tombstone files from deleted objects. It is run on a single Object protocol node and manages cleanup for all of the virtual devices.

The **swift-object-updater** is responsible for updating container listings with objects that were not successfully added to the container when they were initially created, updated, or deleted. Like the object replicator, it is run on a single object protocol node.

The following table shows each of the object services and the set of object protocol nodes on which they need to be executed.

*Table 19. Object services and object protocol nodes*

| Object service | GPFS protocol node |
|---|---|
| **ibmobjectizer** | object-singleton_node [1] |
| **openstack-swift-account** | All |
| **openstack-swift-account-auditor** | object_singleton_node |
| **openstack-swift-account-reaper** | All |
| **openstack-swift-account-replicator** | All |
| **openstack-swift-container** | All |
| **openstack-swift-container-auditor** | object_singleton_node |

*Table 19. Object services and object protocol nodes (continued)*

| Object service | GPFS protocol node |
|---|---|
| **openstack-swift-container-updater** | object_singleton_node |
| **openstack-swift-container-replicator** | All |
| **openstack-swift-object** | All |
| **openstack-swift-object-auditor** | object_singleton_node [2] |
| **openstack-swift-object-replicator** | All |
| **openstack-swift-object-sof** | All [1] |
| **openstack-swift-object-updater** | object_singleton_node |
| **openstack-swift-object-expirer** | object_singleton_node |
| **openstack-swift-proxy** | All |
| **memcached** | All |
| **openstack-keystone** | All [3, 4] |
| **postgresql-obj** | object_database_node [3] |

[1] If unified file and object access is enabled.
[2] If multi-region object deployment is enabled.
[3] If local OpenStack Keystone Identity Service is configured.
[4] Updated to httpd (keystone) on all nodes, if using local authentication.

## Understanding the mapping of OpenStack commands to IBM Spectrum Scale administrator commands

Use this information to map OpenStack commands to IBM Spectrum Scale administrator commands.

In IBM Spectrum Scale, for Object storage, several OpenStack commands have been replaced with IBM Spectrum Scale commands for easy maintenance. This section identifies those commands.

1. Ring Building

   The swift-ring-builder command should only be used to view the object, container, and account ring on any IBM Spectrum Scale protocol node. The user should not directly execute any commands that modify the ring. All ring maintenance operations are handled automatically by the CES infrastructure.

   For example, when a new CES IP address is added to the configuration, all rings are automatically updated to distribute Swift virtual devices evenly across CES IP addresses.

   The master copy of each ring builder file is kept in the IBM Spectrum Scale Cluster Configuration Repository (CCR). Changes made locally to the ring files will be overwritten with the master copy when monitoring detects a difference between the ring file in CCR and the file in `/etc/swift`.

2. Configuration Changes

   The openstack-config command should not be used to update any of the configuration files consumed by IBM Spectrum Scale for Object storage. Furthermore, you should not edit these files directly, but instead modify them using the **mmobj config change** command.

   The master copy of object and related configuration files are kept in the IBM Spectrum Scale CCR. Changes made locally to these config files will be overwritten with the master copy when monitoring detects a difference between the configuration file in CCR and the file in `/etc/swift` or `/etc/keystone`.

## Changing Object configuration values

Use the following information to change the Object configuration values in the Cluster Configuration Repository (CCR).

You can manage the Object configuration data in the Cluster Configuration Repository (CCR). When an object configuration is changed, callbacks on each protocol node will update that node with the change and restart one or more Object services if necessary.

To change the Object configuration, use the **mmobj** command so that the change is made in the CCR and propagated correctly across the Swift cluster.

For more details, see the *mmobj command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Changing the object base configuration to enable S3 API

IBM Spectrum Scale uses Swift3 Middleware for OpenStack Swift, allowing access to IBM Spectrum Scale using Amazon Simple Storage Service (S3) API.

Perform the following steps if S3 API was not enabled as part of the object base configuration:

1. To enable S3 API, run the following command:
   ```
   mmobj s3 enable
   ```
   The system enables S3 API.
2. To verify that S3 API has been enabled, run the following command:
   ```
   mmobj s3 list
   ```
3. To disable S3 API, run the following command:
   ```
   mmobj s3 disable
   ```
   The system disables S3 API.
4. To verify that S3 API has been disabled, run the following command:
   ```
   mmobj s3 list
   ```

## Configuring OpenStack EC2 credentials

The credentials that are used on the Amazon S3 and Elastic Compute Cloud (EC2) APIs are different from the credentials that are used by the OpenStack API. As a result, you must generate these special credentials to use them when accessing the IBM Spectrum Scale OpenStack services.

The credentials are created by the OpenStackClient, a command-line client for OpenStack, that allows the creation and use of access/secret pairs for a user/project pair. This requires the operators to create the access/secret for each user/project pair.

1. Source openrc with the admin credentials.
2. Create EC2 credential by running this command for user-defined blob as a credential:
   ```
   openstack credential create –-type ec2 –-project <project> <user> '{"access": <aws_access_key>, "secret": <aws_secret_key>}'
   ```
   **Note:** Ensure to use Keystone UUIDs rather than names if duplicate user/project names exist across domains. Additionally, the admin users should be able to list and delete access/secrets for a specific user/project.
3. View all EC2 credentials by running this command:
   ```
   openstack credential list
   openstack credential show <credential-id>
   ```
4. You can change your Access Key ID and Secret Access Key if necessary.

It is recommended to have regular rotation of these keys and switching applications to use the new pair.

Change the EC2 credentials by running this command:

```
openstack credential set –type ec2 –data '{"access": <access>, "secret": <secret>}' --project <project> <credential-id>
```

5. Delete the EC2 credentials by running this command:

   ```
   openstack credential delete <credential_id>
   ```

   The following example shows the creation of EC2 credentials using the admin project and the admin user IDs:

   Where openrc contains:

   export OS_AUTH_URL="http://127.0.0.1:35357/v3"

   export OS_IDENTITY_API_VERSION=3

   export OS_AUTH_VERSION=3 export

   OS_USERNAME="admin"

   export OS_PASSWORD="Passw0rd"

   export OS_USER_DOMAIN_NAME=Default

   export OS_PROJECT_NAME=admin

   export OS_PROJECT_DOMAIN_NAME=Default

   source openrc

```
openstack credential create --type ec2 --project admin admin '{"access": "022ABO6E7MXBSH9DHM02", "secret": "pWculUX4JEDGM/LtmEENI/aVmYvHNif5zB+d9+ct"}'
```

You are now ready to connect to the IBM Spectrum Scale Object store using the Amazon S3 API. You can connect with any S3-enabled client.

## Managing OpenStack access control lists using S3 API

The following topic lists the permissions and the known limitations of S3 API.

IBM Spectrum Scale supports S3 access control lists (ACLs) on buckets and objects. These S3 ACLs are stored separately from the ACLs set through the Swift API and the ACLs stored in the file system (NFSv4 or POSIX).

If S3 API is enabled, the default value of **s3_acl** in the `proxy-server.conf` file is `true`. S3 API uses its own metadata for ACL, such as X-Container-Sysmeta-Swift3-Acl, to achieve the best S3 compatibility. However, if S3 API is set to `false`, S3 API tries Swift ACLs, such as X-Container-Read, initially instead of S3 ACLs.

For a user to use S3 API in IBM Spectrum Scale, the user must have a role defined for the swift project. Any role suffices, because for S3 API there is no difference between the `SwiftOperator` role or others.

The owner of a resource is implicitly granted **FULL_CONTROL** instead of just **READ_ACP** and **WRITE_ACP**. This is not a security issue because with **WRITE_ACP**, the owners can grant themselves **FULL_CONTROL** access.

The following table lists the required permissions for S3 operations.

| S3 operation | Required permission |
|---|---|
| PUT object | WRITE permission on bucket or as bucket owner |
| HEAD object | READ permission on object or as object owner |
| GET object | READ permission on object or as object owner |
| DELETE object | WRITE permission on bucket or as bucket owner |
| Get object ACL (GET on ACL subresource) | READ_ACP permission on object or as object owner |

| S3 operation | Required permission |
|---|---|
| Set object ACL (PUT on ACL subresource) | WRITE_ACP permission on object or as object owner |
| Create bucket (PUT) | Any user with a role on the project can create a bucket. |
| HEAD bucket | READ permission on bucket or as bucket owner |
| GET bucket | READ permission on bucket or as bucket owner |
| DELETE bucket | bucket owner |
| Get bucket ACL (GET on ACL subresource) | READ_ACP permission on bucket or as bucket owner |
| Set bucket ACL (PUT on ACL subresource) | WRITE_ACP permission on bucket or as bucket owner |

### Known limitations for S3 API support

- Unauthorized S3 requests are not supported. S3 requests do not contain a reference to the account, and the object server derives the account information from the authorization information. This is not possible for unauthorized requests.
- Specifying S3 ACL grantees by email is not supported.
- Grantees in ACL are not validated. Therefore, any name can be used, even users that do not exist.
- Container or objects created using the swift API are not accessible through S3 API when the `allow_no_owner` configuration flag is set to `false` in `proxy-server.conf`. To change this setting, you can use the following command:

  ```
  mmobj config change --ccrfile proxy-server.conf --section filter:swift3
  --property allow_no_owner --value true
  ```

  The default value of the `allow_no_owner` configuration flag is `true`.
- The POST operation to update metadata has not been implemented.

## Managing object capabilities

You can manage the object capabilities using the following commands.

For an overview of object capabilities, see *Object capabilities* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

- To list all object capabilities available cluster wide, use the **mmobj config list** command as follows:

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
```

  The system displays output similar to the following:

```
file-access-enabled:   true
multi-region-enabled:  true
s3-enabled:            false
```

  You can also list specific object capabilities using the **mmobj config list** command as follows:

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property multi-region-enabled
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities
--property s3-enabled
```

- To enable an object capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property <capability_name> --value true
```

  The system displays output similar to the following:

```
<capability_name>-enabled:   true
```

- To enable the file-access object capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled --value true
```

The system displays output similar to the following:

```
file-access-enabled:   true
```

- To disable an object capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property <capability_name> --value false
```

The system displays output similar to the following:

```
file-access-enabled:   false
```

- To disable the file-access capability, use the **mmobj config change** command as follows:

```
mmobj config change --ccrfile spectrum-scale-object.conf --section capabilities
--property file-access-enabled --value false
```

The system displays output similar to the following:

```
file-access-enabled:   false
```

## Managing object versioning

See the following topics to enable and disable object versioning.

## Enabling object versioning

Perform the following steps to enable object versioning.

1. Alter the proxy-server pipeline by running the following command:

   ```
   mmobj config list --ccrfile proxy-server.conf --section
   pipeline:main
   ```

   The system displays the following output:

   ```
   [pipeline:main]
   pipeline = healthcheck cache formpost tempurl s3token authtoken
   keystoneauth container-quotas account-quotas staticweb bulk slo
   dlo versioned_writes proxy-server
   ```

2. Add filter:versioned_writes:

   ```
   mmobj config list --ccrfile proxy-server.conf --section
   filter:versioned_writes
   ```

   The system displays the following output:

   ```
   [filter:versioned_writes]
   use = egg:swift#versioned_writes
   allow_versioned_writes = true
   ```

## Disabling object versioning

Perform the following steps to disable object versioning.

To disable object versioning across the cluster, run the following command:

```
# mmobj config change --ccrfile proxy-server.conf --section
'filter:versioned_writes' --property allow_versioned_writes --value false
```

The system displays the following output:

```
[filter:versioned_writes]
use = egg:swift#versioned_writes
allow_versioned_writes = false
```

## Creating a version of an object: Example

The following example can be used to understand how to create object versions.

1. Create a container with the **X-Versions-Location** header or add the header to an existing container. In this example, *version_container* is the container that holds old versions of objects and *container1* is a new or existing container for which object versioning is to be enabled.

   ```
   swift post -H "X-Versions-Location: version_container" container1
   ```

2. Run **swift stat** on *container1* to check that **X-Versions-Location** header is applied:

   ```
   swift stat container1
              Account: AUTH_f92886c4e3a347a18c29bae581b36788
            Container: container1
              Objects: 0
                Bytes: 0
           Read ACL:
          Write ACL:
            Sync To:
           Sync Key:
       Accept-Ranges: bytes
     X-Storage-Policy: Policy-0
           Connection: keep-alive
          X-Timestamp: 1468226043.18746
           X-Trans-Id: tx8d17476a914a40d781b0a-0057835a01
        Content-Type: text/plain; charset=utf-8
   X-Versions-Location: version_container
   ```

3. If *version_container* does not exist, create a new container:

   ```
   swift post version_container
   ```

4. Run **swift list** at the account level to check that both containers are created successfully:

   ```
   swift list
   container1
   version_container
   ```

5. Upload an object to *container1*:

   ```
   swift upload container1 ImageA.jpg
   ```

6. Upload the second version of the object:

   ```
   swift upload container1 ImageA.jpg
   ```

7. Upload the third version of the object:

   ```
   swift upload container1 ImageA.jpg
   ```

8. Run **swift list** on the container to view the stored object:

   ```
   swift list container1
       ImageA.jpg
   ```

   **Note:** The *container1* container contains only the latest version of the objects. The older versions of object are stored in *version_container*.

9. Run **swift list** on *version_container* to see the older versions of the object:

   ```
   swift list version_container
    00aImageA.jpg/1468227497.47123
    00aImageA.jpg/1468227509.48065
   ```

10. To delete the latest version of the object, perform the DELETE operation on the object:

    ```
    swift delete container1 ImageA.jpg
    ImageA.jpg
    (deleted latest/third version)

    # swift list container1
    ImageA.jpg
    (Second version is now the latest version)

    # swift list version_container
    00aImageA.jpg/1468227497.47123
    (Initial version of the object)
    ```

## Mapping of storage policies to filesets

For every storage policy created using the **mmobj policy create** command, one fileset is created or reused.

After a storage policy is created, you can specify that storage policy while creating new containers to associate that storage policy with those containers. When objects are uploaded into a container, they are stored in the fileset that is associated with the container's storage policy. For every new storage policy, a new object ring is created. The ring defines where objects are located and also defines multi-region replication settings.

The name of the fileset can be specified optionally as an argument of the **mmobj policy create** command. An existing fileset can be used only if:

- It is not being used for an existing storage policy.
- It is a part of the object file system
- Its junction path is not nested to other filesets.

If even one of these prerequisites is missing, the **mmobj policy create** command fails. Otherwise, the fileset is used and the softlinks for the devices that are given to the ring builder point to it. If no fileset name is specified with the **mmobj policy create** command, a fileset is created using the policy name as a part of the fileset name with the prefix `obj_`.

For example, if a storage policy with name Test is created and no fileset is specified, a fileset with the name `obj_Test` is created and is linked to the base file system for object:

```
<object base filesystem mount point>/obj_Test/<n virt. Devices>
```

**Attention:** For any fileset that is created, its junction path is linked under the base file system for object. The junction path should not be changed for a fileset that is used for a storage policy. If it is changed, data might be lost or it might get corrupted.

To enable swift to work with the fileset, softlinks under the given devices path in `object-server.conf` are created:

```
<devices path in object-server.conf>/<n virt. Devices>
<object base filesystem mount point>/obj_Test/<n virt. Devices>
```

## Administering storage policies for object storage

Use the following information to create, list, and change storage policies for object storage.

Before creating a storage policy with the file-access (unified file and object access) function enabled, the file-access object capability must be enabled. For more information, see "Managing object capabilities" on page 188 and *Object capabilities* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

- To create a new storage policy with the unified file and object access feature enabled, run the following command:

  ```
  mmobj policy create sof-policy --enable-file-access
  ```

  The system displays output similar to this:

  ```
  [I] Getting latest configuration from ccr
  [I] Creating fileset /dev/gpfs0:obj_sof-policy
  [I] Creating new unique index and build the object rings
  [I] Updating the configuration
  [I] Uploading the changed configuration
  ```

- To list storage policies for object storage with details of functions available with those storage policies, 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"
```

- To change a storage policy for object storage, run the following command:

  ```
  mmobj policy change
  ```

- 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.

For more information about the **mmobj policy** command, see *mmobj command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Creating storage policy for object compression

Use the following information to create a storage policy with the compression function enabled and to create a storage policy with the compression schedule defined.

- To create a storage policy with the compression function enabled, use the `--enable-compression` option with the **mmobj policy create** command as follows:

  ```
  mmobj policy create CompressionTest --enable-compression --compression-schedule "MM:HH:dd:ww"
  ```

- To create a storage policy with the compression function enabled and a compression schedule defined, use the `--enable-compression` and the `--compression-schedule` options with the **mmobj policy create** command as follows:

  ```
  mmobj policy create CompressionTest --enable-compression --compression-schedule "MM:HH:dd:ww"

  where
  MM = 0-59 minutes
  HH = 0-23 hours
  dd = 1-31 day of month
  ww = 0-7 (0=Sun, 7=Sun) day of week
  ```

  – Use * for specifying every instance of a unit. For example, dd = * means that the job is scheduled to run every day.
  – Comma separated lists are allowed. For example, dd = 1,3,5 means that the job is scheduled to run on every 1st, 3rd, 5th of a month.
  – If ww and dd both are specified, the union is used.
  – Specifying a range using - is not supported.
  – Empty values are allowed for dd and ww. If empty, dd and or ww are not considered.
  – Empty values for mm and hh are treaded as *.

  In the following example, the compression job has been scheduled to run at 23.50 every day:

  ```
  mmobj policy create CompressionTest --enable-compression --compression-schedule "50:23:*:*"
  ```

  Every object stored using a storage policy that has compression enabled is compressed according to the specified schedule. There is no need to decompress an object in advance of a get request or any other object request. IBM Spectrum Scale automatically returns the decompressed object.

  **Note:** The download performance of objects in a compressed container is reduced compared to the download performance of objects in a non-compressed container.

  **Note:** The same compression functionality and restrictions apply to object compression and file compression.

**Related concepts**:

File compression

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

## Creating storage policy for object encryption

Use the following information to create a storage policy with the encryption function enabled and to create a storage policy with the encryption schedule defined.

To create a storage policy with the encryption function enabled, use the **mmobj policy create** command as follows:

```
mmobj policy create PolicyName [-f FilesetName] [-i MaxNumInodes]
{[--enable-compression --compression-schedule CompressionSchedule]}
[--enable-file-access] {[--enable-encryption --encryption-keyfile EncryptionKeyFileName [–force-rule-add]]}

where
PolicyName is the name of the storage policy that must be created.
FilesetName is the fileset name the created storage policy must use. This parameter is optional.
MaxNumInodes specifies the Inode limit for the new inode space. This parameter is optional.
enable-compression enables a compression policy. This parameter is optional.
CompressionSchedule specifies the compression schedule. This parameter is optional.
enable-file-access enables a file-access policy. This parameter is optional.
enable-encryption enables an encryption policy. This parameter is optional.
EncryptionKeyFileName specifies the encryption key file (full qualified).
This parameter is optional.
force-rule-add specifies whether to add and establish the rule if other rules exist already.
This parameter is optional.
```

The –force-rule add parameter is used the decide the following cases whether to establish the GPFS policy rules.

- `--force-rule-add`

  is not set:

  – During create policy we check whether a GPFS policy rule is already established.
  – If yes, the new encryption rule will not be established but will be printed on the command screen.
  – If there is no previous policy rule established, the new encryption rule will be established and printed on the command screen.

- `--force-rule-add`

  is set:

  – During create policy we check whether a GPFS policy rule is already established.
  – If yes, the new encryption rule will be added to the already established rules and the GPFS policy for the filesystem is updated. The new encryption rule will be printed on the command screen.
  – If there is no previous policy rule established, the new encryption rule will be established and printed on the command screen.

During command execution the encryption policy and rule will be created. A GPFS policy rule file will be created and used to establish the policy rule.
Policy Rule File:

- filename = `/var/mmfs/ces/policyencryption.rule`

**Note:** The filename is auto-generated.
After the encryption policy is created, depending on the –force-rule-add parameter, as a user information the new encryption policy will be printed on the command screen.

If an error on creating the encryption part occurs, the local cleanup function is called to remove the just created fileset and exit the CLI mmobj policy create script. The existing rules and policies are not changed.

**Note:** The same encryption functionality and restrictions apply to object encryption as they apply for file encryption.

## Adding a region in a multi-region object deployment

Perform the following steps to add a region in a multi-region object deployment environment.

In the command examples, Europe is the first region and Asia is the second region.

1. Export the information of the first region to a file by using the **mmobj multiregion export** command.
   For example:
   ```
   [europe]# mmobj multiregion export --region-file /tmp/multiregion_europe.dat
   ```
2. Copy the file manually to the second region.
   For example:
   ```
   [europe]# scp /tmp/multiregion_europe.dat asia:/tmp
   ```
3. From the second region, join the multi-region environment as follows:
   a. Use the file generated in the first region while deploying object on the second region by using the **mmobj swift base** command.
      For example:
      ```
      [asia]# mmobj swift base -g /mnt/gpfs0 --cluster-hostname gpfs-asia --admin-password Passw0rd
      -i 100000 --admin-user admin --enable-s3 \
        --enable-multi-region  --remote-keystone-url  http://gpfs-asia:35357/v3
       --join-region-file /tmp/multiregion_europe.dat \
              --region-number 2 --configure-remote-keystone
      ```
      This step installs the object protocol in the 2nd region and joins the 1st region. Additional devices are added to the primary ring files for this region.
4. Export the ring file data of the second region.
   For example:
   ```
   [asia]# mmobj multiregion export --region-file /tmp/multiregion_asia.dat
   ```
5. Copy the file manually to the first region.
   For example:
   ```
   [asia]# scp /tmp/multiregion_asia.dat europe:/tmp
   ```
6. In the first region, update the local ring files with the configuration of the second region.
   For example:
   ```
   [europe]# mmobj multiregion import --region-file /tmp/multiregion_asia.dat
   ```
   This step reads in the ring files which are updated with the information of the second region. This update ensures that the data of the second region contains a new region and therefore replaces the associated ring files in the first region with the ones from the second region.

**Note:**

Now the two clusters have been synced together and can be used as a multi-region cluster. Objects can be uploaded and downloaded from either region. If the installation of the second region specified the **--configure-remote-keystone** flag, a region-specific endpoint for the object-store service for the 2nd region is created in Keystone.

The regions need to be synced in the future any time region-related information changes. This includes changes in the set of CES IP addresses (added or removed) or if storage policies were created or deleted

within a region. Changes that affect the `swift.conf` file or ring files need to be synced to all regions. For example, adding additional CES addresses to a region causes the ring files to be rebuilt.

7. In the second region, add CES addresses and update other clusters.

   For example:

   ```
   [asia]# mmces address add --ces-ip asia9
   ```

   This steps adds an address to the CES IP pool. This also triggers a ring rebuild which changes the IP-to-device mapping in the ring files.
8. Export the ring data so the other clusters in the region can be updated with the new IPs from the second region.

   For example:

   ```
   [asia]# mmobj multiregion export --region-file /tmp/multiregion_asia.dat
   ```
9. Copy the file manually to the first region.

   For example:

   ```
   [asia]# scp /tmp/multiregion_asia.dat europe:/tmp
   ```
10. In the first region, update with changes for the new second region address in the ring.

    For example:

    ```
    [europe]# mmobj multiregion import --region-file /tmp/multiregion_asia.dat
    ```

    This step imports the changes from the second region. When this is complete, a checksum is displayed which can be used to determine when regions are synchronized together. By comparing it to the one printed when the region data was exported, you can determine that the regions are synchronized when they match. In some cases, the checksums do not match after import. This is typically due to some local configuration changes on this cluster which are not yet synced to the other regions. If the checksums do not match, then this region's configuration needs to be exported and imported into the other region to sync them.

## Administering a multi-region object deployment environment

Use the following information to administer a multi-region object deployment environment.

A multi-region environment consists of several independent storage clusters linked together to provide unified object access. Configuration changes in one cluster which affect the multi-region environment are not automatically distributed to all clusters. The cluster which made the configuration change must export the relevant multi-region data and then the other regions must import that data to sync the multi-region configuration. Changes which affect multi-region are:

- Changes to the CES IP pool, such as adding or deleting addresses, which affect the ring layout.
- Changes to the object services ports used for the account, container, and object servers (ports 6200-6202).
- Creation, deletion, or modification of storage policies.
- Changes to the `swift.conf` configuration file

Use the following commands to manage the configuration of the multi-region environment:

- To export the data for the current region so that it can be integrated into other regions, use the following command. The *RegionData* file created can be used to update other regions:

  ```
  mmobj multiregion export --region-file RegionData
  ```

  The *RegionData* file is created and it contains the updated multi-region information.
- To import the multi-region data to sync the configuration, use the following command. The *RegionData* must be the file created from the **mmobj multiregion export** command:

  ```
  mmobj multiregion import --region-file RegionData
  ```

As part of the export/import commands, a region checksum is printed. This checksum can be used to ensure that the regions are in sync. If the checksums match, then the multi-region configuration of the

clusters match. In some cases, the checksums do not match after import. This is because the cluster performing the import had local configuration changes which had not been synced with the other regions. For example, a storage policy was created but the multi-region configuration was not synced with the other regions. When this happens the import command prints a message that the regions are not fully in sync because of the local configuration and that the region data must be exported and imported to the other regions. Once all regions have matching checksums, the multi-region environment is in sync.

An existing region can be completely removed from the multi-region environment. This action permanently removes the region configuration, and the associated cluster cannot rejoin the multi-region environment.

The cluster of the removed region needs to disable object services since it will not be usable as a standalone object deployment.

- To remove a previously defined region from the configuration, use the following command:

  ```
  mmobj multiregion remove --remove-region-number RegionNumber
  ```

  The remove command must be run from a different region than the one being removed. The cluster associated with the removed region must cleanup object services as appropriate with the **mmces service disable OBJ -a** command to uninstall object services.
- You can display the current multi-region information using the following command:

  ```
  mmobj multiregion list
  ```

# Unified file and object access in IBM Spectrum Scale

Unified file and object access allows use cases where you can access data using object as well as file interfaces. Use the following information to manage unified file and object access including identity management modes for unified file and object access, authentication for unified file and object access, and objectization service schedule.

**Important:** In a unified file and object access environment, object ACLs apply only to accesses through the object interface and file ACLs apply only to accesses through the file interface.

For example: If user Bob ingests a file from the SMB interface and user Alice does not have access to that file from the SMB interface, it does not mean that Alice does not have access to the file from the object interface. The access rights of Alice for that file or object from the object interface depends on the ACL defined for Alice on the container in which that file or object resides.

## Identity management modes for unified file and object access

The following section gives information about the two identity management modes for unified file and object access: local mode and unified mode. This section also describes how to configure these modes for a system.

Unified file and object access comprises the following two modes:

- **local_mode**: Separate identity between object and file (Default mode)
- **unified_mode**: Shared identity between object and file

The mode is represented by the **id_mgmt** configuration parameter in the `object-server-sof.conf` file:

```
id_mgmt = local_mode | unified_mode
```

You can change this parameter by using the **mmobj config change** command. For more information, see "Configuring authentication and setting identity management modes for unified file and object access" on page 206.

**Note:**

- Only one mode can be effective at a given time and it needs to be configured by the administrator for the entire system. **id_mgmt = local_mode** is the default setting.
- If you plan to use unified_mode, the authentication mechanism for file and object must be the same. If you set **id_mgmt** to unified_mode and the file authentication and object authentication are not common, then the ID resolution of the users will not work correctly. This will lead to either object not being created with 503 error* return code or object being created with improper user ID. Therefore, it is very important that the administrators ensure that a common authentication with appropriate ID mapping is configured for file and object.

  * If you are using swift client, instead of 503, you might get an error similar to the following:

  ```
  'put_object('container_name', 'object_name', ..) failure and no ability to reset contents for reupload.'
  ```

| For more information about validating the ID mapping, see *Validating shared authentication ID mapping* in *IBM Spectrum Scale: Administration Guide*

## local_mode - separate identity between object and file

The following points should be considered when planning to use local_mode identity management.

- Use-case for unified file and object access in local_mode:
  - Data created from the object interface is available for application to run analytics using the file interface, where ownership of files is not essential.
  - Data created from the file interface is accessible from the object interface after objectization of those files.
- To address this use case, object authentication setup is independent of file authentication setup. Although, you can set up object and file authentication from a common authentication server in case of AD or LDAP.
- Objects created or updated using the object interface are owned by the swift user. Application processing the object data from file interface need the required file ACL to access the object data.
- Data updated from the file interface after objectization is available for object access.
- Containers created with a unified file and object access policy that are exposed as export points need appropriate ACLs set as needed by SMB, NFS, and POSIX.
- If the object already exists, existing ownership of the corresponding file is retained if retain_owner is set to yes in object-server-sof.conf. For more information, see "Configuration files for IBM Spectrum Scale for object storage" on page 217.
- Retaining ACL, extended attributes (xattrs), and Windows attributes (winattrs): If the object is created or updated over existing file then existing file ACL, xattrs, and winattrs are retained if retain_acl, retain_xattr, and retain_winattr are set to yes in object-server-sof.conf. For more information, see "Configuration files for IBM Spectrum Scale for object storage" on page 217.

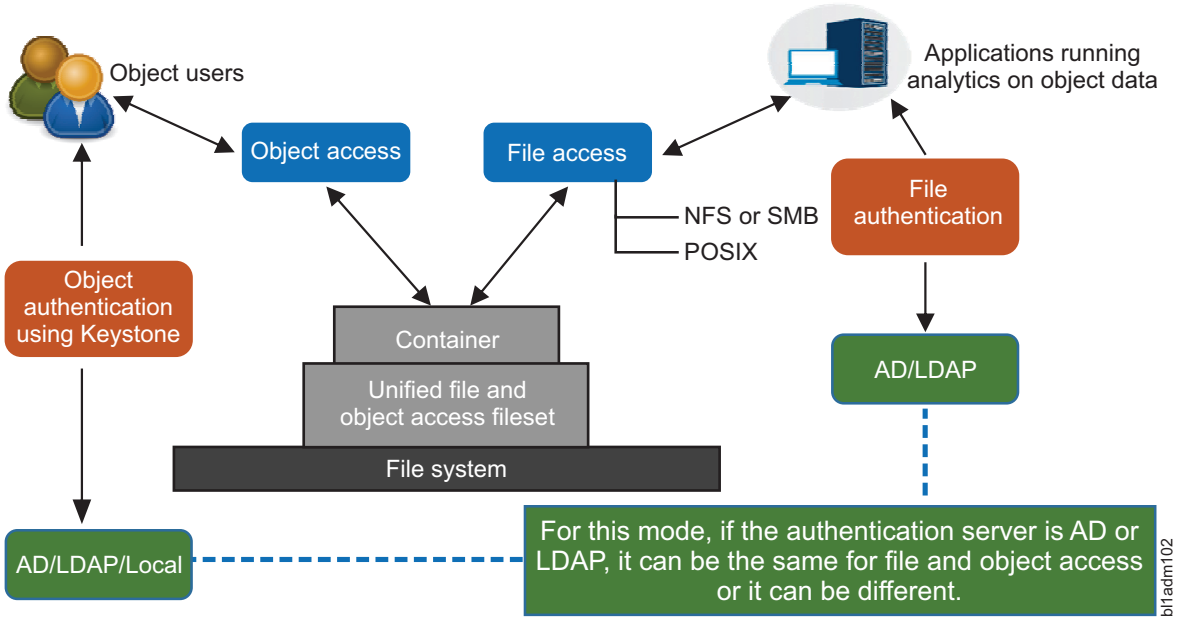



*Figure 3. `local_mode` - separate identity between object and file*

## unified_mode - shared identity between object and file

The following points should be considered when using unified_mode identity management.

- Users from object and file are expected to be common and coming from the same directory service (only AD+RFC 2307 or LDAP).

  **Note:** If your deployment uses only SMB-based file interface for file access and file authentication is configured with Active Directory (AD) with Automatic ID mapping, unified file and object access can be used, assuming that object is configured with the same AD domain.
- Ownership: Object created from the object interface is owned by the user doing the object PUT operation.
- If the object already exists, existing ownership of the corresponding file is retained if `retain_owner` is set to `yes` in `object-server-sof.conf`. For more information, see "Configuration files for IBM Spectrum Scale for object storage" on page 217.
- Authorization: Object access follows the object ACL semantics and file access follows the file ACL semantics.
- Retaining ACL, extended attributes (xattrs), and Windows attributes (winattrs): If the object is created or updated over existing file then existing file ACL, xattrs, and winattrs are retained if `retain_acl`, `retain_xattr`, and `retain_winattr` are set to `yes` in `object-server-sof.conf`. For more information, see "Configuration files for IBM Spectrum Scale for object storage" on page 217.
- When a user does a PUT operation for an object over an existing object or does a PUT operation for a fresh object over a nested directory, no explicit file ACL is set for that user. This means that it is possible that in some cases, the user might not have access to that file from the file interface even though the user has access from the object interface. This is done to prevent changing of the file ACL from the object interface to maintain file ACL semantics. In these cases, if the user is required to have permission to access the file also, explicit file ACL permission need to be set from the file interface.

  For example: If user Bob performs a PUT operation for an object over an existing object (object maps to a file) owned by user Alice, Alice continues to own the file and there is no explicit file level ACL that is set for Bob for that file. Similarly, when Bob performs a PUT operation for a new object inside a subdirectory (already created by Alice), no explicit file ACL is set on the directory hierarchy for Bob.

Bob does not have access to the object from the file interface unless there is an appropriate directory inherence ACL that is set. To summarize, the object ingest does not change any file ACL and vice versa.

*Table 20. Object input behavior in `unified_mode`.*

**Note:** In the scenarios listed in the following table, the operations are being done by user **Bob** from the object interface. The instances owned by user Alice imply that the file or directory ownership maps to user **Alice** from the file side. Also, it is assumed that the `retain_owner`, `retain_acl`, `retain_xattr`, and `retain_winattr` parameters are set to `yes` in `object-server-sof.conf`.

| Operation from SWIFT interface on object or container | Ownership result on corresponding file or directory | | ACL, xattr, and winattr retention behavior on corresponding file or directory | |
|---|---|---|---|---|
| | File | Directory | File | Directory |
| Bob does a PUT operation for a new object | The ownership of the file is set to Bob | NA | Default GPFS ACLs are set | NA |
| Bob does a PUT operation for a new container | NA | The ownership of the directory is set to Bob | NA | Default GPFS ACLs are set |
| Bob does a PUT operation for an object that is already present and is owned by Alice | The ownership of the file continues to be with Alice. Bob is not given any file ACL explicitly. | No changes in the ownership of the parent directory | Existing ACL, file xattrs, and file winattrs are retained** | NA |
| Bob does a POST operation (update metadata) of existing object owned by Alice | The ownership of the file continues to be with Alice. Bob is not given any file ACL explicitly | NA | Existing ACL, file xattrs, and file winattrs are retained** | NA |
| Bob does a POST operation (update metadata or ACL) of existing container owned by Alice | NA | The ownership of the directory continues to be with Alice. Bob is not given any directory ACL | NA | NA |
| GET/DELETE/HEAD | No impact | | | |

**Note:** **Unified file and object access retains the extended attributes (xattr), Windows attributes (winattrs) and ACL of the file if there is a PUT request from an object over an existing file. However, security or system namespace of extended attributes and other IBM Spectrum Scale extended attributes such as immutability, pcache, etc. are not retained. Swift metadata (`user.swift.metadata`) is also not retained and it is replaced according to object semantics which is the expected behavior.

## Advantages of using `unified_mode`

IBM Spectrum Scale offers various features that leverage user identity (UIDs or GIDs). With `unified_mode`, you can use these features seamlessly across file and object interfaces.

- **Unified access to object data:** User can access object data using SMB or NFS exports using their AD or LDAP credentials.
- **Quota:** GPFS quota for users that work on UID or GID can be set such that they work for the file as well as object interface.

  Example: User A can have X quota on a unified access fileset assigned using GPFS quota commands which can hold true for all data created by the user from the file or the object interface.

  For more information, see the *Quota related considerations for `unified_mode`* section.

- **ILM:** Tiering of user specific data leveraging UID or GID.

  Example 1: The UID and GID file attributes can be used to create an ILM placement policy to place the files owned by the Gold customers in faster storage pools and retain the files in the pools even when the pool storage starts reaching the threshold. The UID and GID file attributes can also be used to create a migration ILM policy so that, when the pool reaches its storage threshold, all files older than 30 days are moved to a slower storage pool except the ones owned by the Gold customers.

  Example 2: After a user has left the organization, the UID of the user can be used to migrate the data and retain it on the archive tape for as long as defined as defined by the ILM policies.
- **Backup:** Backup of user specific data leveraging UID or GID.

  Example: UID and GID file attributes in the policy rules that are defined for the mmbackup command can be used to regularly back up the data of selective users.

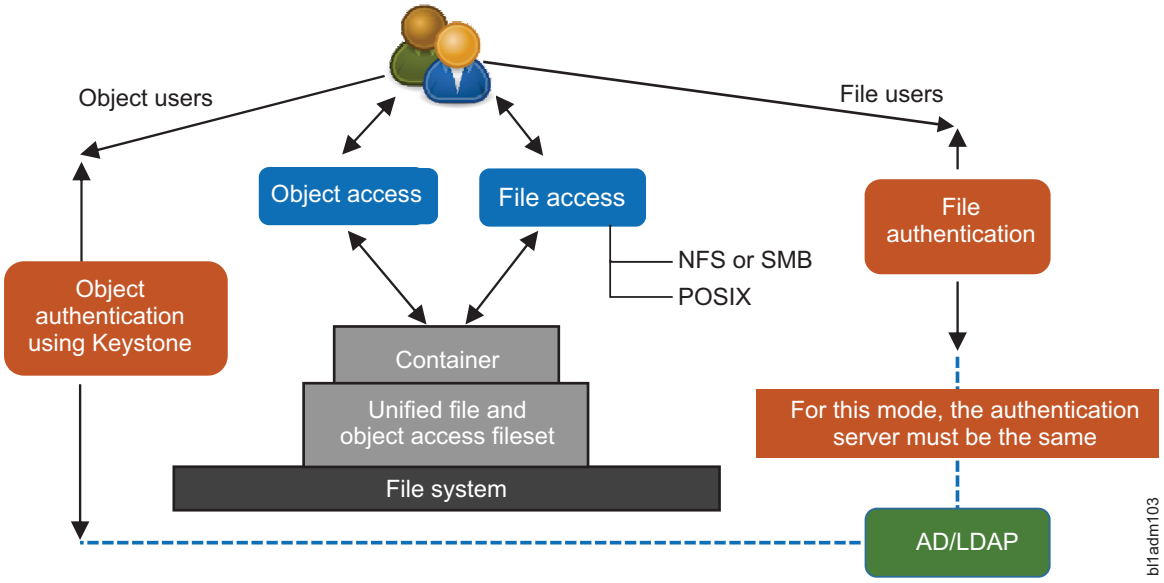



*Figure 4. `unified_mode` - unified identity between object and file*

### Quota related considerations for `unified_mode`

There are three types of quotas that need to be considered:

- Quota for a user set using file system commands for that fileset which is set using User ID or Group ID. This quota represents the size in bytes up to which the user can create data on a given fileset. This is tracked at the file system level.
- Container quota: This is the size in bytes or number of objects that can be stored in a container regardless of the user making the upload (PUT) request. For more information on container quotas, see OpenStack documentation of container quotas.
- Account quotas: This is the size in bytes or number of objects that can be stored in an account regardless of the user making the upload (PUT) request. For more information on account quotas, see OpenStack documentation of account quotas.

The fileset quotas and container level quotas as well as fileset quotas and account quotas are independent of each other.

In some cases, the fileset quota should be cumulative of all the containers' quotas hosted over it, though it is not mandatory. When both the quotas at the fileset level as well as at the container quota level are set, and if the fileset quota is reached, no more object data can be input on any of the containers hosted

by that fileset, despite of the container quota not being reached. Hence, when you plan to use both file and object quotas, it is important to understand these details.

The objectization process does not take into account the container quota and the account quota. This means that there might be a scenario where a container can host more data than the container quota associated with it especially when the **ibmobjectizer** service has objectized files as objects.

For example, consider that:

- You want to have a total of 1 TB of data allocated for file and object access.
- You want each user to have an overall quota from the file as well as the object interface to be 10 GB.
- You have a pre-defined set of 100 containers which are enabled for object and file access (using the storage policy for object storage) and users access to different containers is dependent on the container ACLs.

In this case, quotas are set as follows:

1. Set the fileset quota associated with the file access policy to 1 TB.
2. Set the user quota on that fileset to 10 GB.
3. Set the container quota to the required level. However, setting it more than fileset quota cannot be honored until fileset quota is increased or unset.

In this example scenario, note that the object access will be restricted if either the user quota or the fileset quota is reached, even though the container quota is not reached.

## Authentication in unified file and object access

The following section gives information about how file authentication and object authentication are configured for different identity management modes.

### Authentication configuration in local_mode: separate identity between object and file

In this mode, all the objects created continue to be owned by the `swift` user, that is an administrator under whose context the object server runs on the system. Because in this mode there is no ID mapping of objects to user ID, object authentication can be configured to any supported authentication schemes and file authentication can continue to be configured to any supported authentication scheme.

For supported authentication schemes, see the *Authentication support matrix* table in the *Authentication considerations* topic in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

### Authentication configuration in unified_mode: shared identity between object and file

This mode allows objects and files to be owned by the users' UID and the corresponding GID that created them. This mode mandatorily requires both the object protocol and the file protocol to be configured with the same authentication scheme. The supported authentication schemes for the unified mode are:

- AD for Authentication + RFC 2307 for ID mapping
- LDAP for authentication as well as for ID mapping

**Note:** User-defined authentication is not supported with both the identity management modes.

## Validating shared authentication ID mapping

Perform the following steps to validate shared authentication ID mapping.

1. List the authentication details on IBM Spectrum Scale by running the **mmuserauth service list** command. The system displays the output similar to the following:

```
FILE access configuration : LDAP
PARAMETERS VALUES
-------------------------------------------------
ENABLE_SERVER_TLS false
ENABLE_KERBEROS false
USER_NAME cn=manager,dc=sonasldap,dc=com
SERVERS 9.118.37.234
NETBIOS_NAME deepakcluster
BASE_DN dc=sonasldap,dc=com
USER_DN dc=sonasldap,dc=com
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 configuration : LDAP
PARAMETERS VALUES
-------------------------------------------------
ENABLE_ANONYMOUS_BIND false
ENABLE_SERVER_TLS false
ENABLE_KS_SSL false
USER_NAME cn=manager,dc=sonasldap,dc=com
SERVERS 9.118.37.234
BASE_DN dc=sonasldap,dc=com
USER_DN dc=sonasldap,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 userr
```

2. Ensure that the file authentication type and the object authentication type are the same. The valid values are AD and LDAP. The following are examples of file authentication and object authentication types:

```
FILE access configuration : LDAP
OBJECT access configuration : LDAP
```

3. Configure the file authentication and the object authentication against the same server:

```
FILE : SERVERS 9.118.37.234
OBJECT : SERVERS 9.118.37.234
```

   **Note:** If there are multiple domain controllers in AD, the values might not match. The administrator must ensure that the server is referring to same user authentication source.

4. Ensure that the object users are receiving the correct UIDs and GIDs from the authentication source. In the following example, the userr object user, is being used:

```
cat openrc
# Thu May 26 19:04:37 IST 2016
export OS_AUTH_URL="http://127.0.0.1:35357/v3"
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_VERSION=3
export OS_USERNAME="userr"
export OS_PASSWORD=""
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_NAME=Default
```

5. Ensure that the object user is correctly resolved on all the protocol nodes and the same UID and GID are listed. In the following example, the UID and GID of the userr object user has been listed:

```
id userr
uid=1101(userr) gid=1000(testgrp) groups=1000(testgrp),1002(testgrp2)
```

## The objectizer process

The objectization process converts files ingested from the file interface on unified file and access enabled container path to be available from the object interface. The name of the service that does this is **ibmobjectizer**.

When new files are added from the file interface, they need to be visible to the Swift database to show correct container listing and container or account statistics.

The **ibmobjectizer** service ensures synchronization between the file metadata and the object metadata at predefined time interval that ensures accurate container and account listing. The **ibmobjectizer** service identifies new files added from the file interface and adds the Swift system metadata to them so that they are objectized. The **ibmobjectizer** service then determines its container and account databases and adds a new object entry to those. It also identifies files deleted from file interface and deletes their corresponding entries from container and account databases.

This is particularly useful in setups where data is ingested using legacy file interface based devices such as medical and scientific devices and it needs to be stored and accessed over cloud using the object interface.

The **ibmobjectizer** service is a singleton and it is started when object is enabled and the file-access object capability is set. However, the **ibmobjectizer** service starts objectization only when there are containers with unified file and object access storage policies configured and the file-access object capability is set.

To identify the node on which the **ibmobjectizer** service is running, use the **mmces service list --verbose** command.

For example, if you have a cluster that has gpfsnode3 as the object singleton node, run the following command:

```
mmces service list --verbose -a | grep ibmobjectizer
```

The system displays the following output:

```
gpfsnode3:       OBJ:ibmobjectizer                          is running
```

**Attention:** If object services on the singleton node are stopped by the administrator manually, objectization is stopped across the cluster. Therefore, manually stopping services on a singleton node must be planned carefully after understanding its impact.

For information on limitations on the objectizer process, see "Limitations of unified file and object access" on page 213.

**Related concepts**:

"Understanding and managing Object services" on page 183
Use the following information to manage services related to IBM Spectrum Scale for object storage.

**Related tasks**:

"Setting up the objectizer service interval" on page 206
Take the following steps to set up the objectizer service interval.

**Related reference**:

"Configuration files for IBM Spectrum Scale for object storage" on page 217
Use the following information to manage options in configuration files that are used for IBM Spectrum Scale for object storage including the unified file and object access feature. These configuration files are

located in the /etc/swift directory.

## File path in unified file and object access

One of the key advantages of unified file and object access is the placement and naming of objects when stored on the file system.

Unified file and object access stores objects following the same path hierarchy as the object's URL. In contrast, the default object implementation stores the object following the mapping given by the ring, and its final file path cannot be determined by the user easily. For example, an object with the following URL is stored by the two systems as follows:

- **Example object URL:** https://swift.example.com/v1/acct/cont/obj
- **Path in default object implementation:** /ibm/gpfs0/object_fileset/o/z1device108/objects/7551/125/75fc66179f12dc513580a239e92c3125/75fc66179f12dc513580a239e92c3125.data
- **Path in unified file and object access:** /ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_763476384728498323747/cont/obj

In this example, it is assumed that the object is configured over the /ibm/gpfs0 file system with the default object located on the object_fileset fileset and the unified file and object access data is located under the obj_sofpolicy1 fileset. s69931509221z1device1 is auto-generated based on the swift ring parameters and AUTH_763476384728498323747 is auto-generated based on the account ID from keystone.

**Attention:** Do not unlink object filesets including the unified file and object access enabled filesets.

### Determining the POSIX path of a unified file and object access enabled fileset

Use the following steps for determining the POSIX path of a unified file and object access enabled fileset.

1. List all storage policies for object.

```
mmobj policy list

Index       Name             Default Deprecated Fileset              Functions
-------------------------------------------------------------------------------------------
0           SwiftDefault     yes                object_fileset
13031510160 sof-policy1                         obj_sof-policy1      file-and-object-access
13031510260 CompressionTest          yes        obj_CompressionTest  compression
13031510290 CompressionDebug         yes        obj_CompressionDebug compression
13031511020 CompressionNew                      obj_CompressionNew   compression
```

2. In the fs0 file system, note the index and fileset name for the policy you are interested in and use the **mmlsfileset** command to determine the junction point.

```
mmlsfileset fs0 | grep obj_sof-policy1
obj_sof-policy1          Linked    /ibm/fs0/obj_sof-policy1
```

The swift ring builder creates a single virtual device for unified file and object access policies, named with storage policy index number, which is also the region number, starting with s and appended with z1device1.

```
s13031510160z1device1
```

3. List the swift projects and identify the one you are interested in working with.

```
~/openrc
openstack project list
+----------------------------------+---------+
| ID                               | Name    |
+----------------------------------+---------+
| 73282e8bca894819a3cf19017848ce6b | admin   |
| 1f78f58572f746c39247a27c1e0e1488 | service |
+----------------------------------+---------+
```

4. Construct the account name by appending the project ID with AUTH_ or substitute the correct project prefix if you have customized this. For the admin project, use:

```
AUTH_73282e8bca894819a3cf19017848ce6b
```

The full path to the unified file and object access containers is the concatenation of the fileset linkage, the virtual device name, and the account name:

```
/fileset linked path/s/region number/z1device1/AUTH_account id/
```

An example of the file path is:

```
/ibm/fs0/obj_sof-policy1/s13031510160z1device1/AUTH_73282e8bca894819a3cf19017848ce6b/
```

**Note:** To substitute the correct project prefix, see *Managing object users, roles, and projects* in *IBM Spectrum Scale: Administration Guide*.

5. List the containers defined for this account.

```
ls /ibm/fs0/obj_sof-policy1/s13031510160z1device1/AUTH_73282e8bca894819a3cf19017848ce6b/
new1    fifthcontainer  RTC73189_1  RTC73189_3  RTC73189_5  RTC73189_7  sixthcontainer
```

The following is an example of determining the POSIX path of a unified file and object access enabled fileset from the command line with one rather complex command. This assumes that your GPFS file system device is `gpfs0`, and also that you are interested in the first storage policy listed. If this is not the case, update the script accordingly.

Commands used:

```
# openstack project list
# swift capabilities
```

Assumptions:

```
# File system name is is gpfs0
# Project/Account name is admin
# first [1] SoF policy is considered
```

```
echo $(find $(mmlsfileset gpfs0 | grep $(perl -e '$p=`swift capabilities
| grep policies:`;$p=~s/:/=>/g;eval"\$v= {".$p."}";print$v->{policies}[1]{name};')
| awk '/ / {print $3}') -name AUTH_$(openstack project list | awk '/ admin / {print $2}'))
```

# Administering unified file and object access

Use the following information to administer unified file and object access in your IBM Spectrum Scale setup.

## Enabling the file-access object capability

Before you can use unified file and object access, you must enable the file-access object capability on the whole cluster.

- Enable the file-access object capability using the **mmobj config change** as follows.

```
mmobj file-access enable
```

- Verify that the file-access object capability is enabled using the **mmobj config list** as follows.

```
mmobj config list --ccrfile spectrum-scale-object.conf --section capabilities --property file-access-enabled
```

The system displays output similar to the following:

```
file-access-enabled = true
```

## Starting and stopping the ibmobjectizer service

This topic lists the commands to start and stop the ibmobjectizer service.

The ibmobjectizer service can be started and stopped by running the following commands.

**Note:** The ibmobjectizer service starts when **file-access-enabled** is set to true.

- To start the ibmobjectizer service when it has stopped, type `mmobj file-access enable`.
- To stop the ibmobjectizer service, type `mmobj file-access disable --objectizer`.
- To check the service status of the ibmobjectizer service, type `mmces service list -v -a | grep ibmobjectizer .`

## Setting up the objectizer service interval

Take the following steps to set up the objectizer service interval.

The default interval between the completion of an objectizer cycle and the starting of the next cycle is 30 minutes. However, this needs to be planned properly based on the following:

- The frequency and the number of new file ingestions that you are expecting to be objectized.
- The number of protocol nodes you have deployed.
- How quickly you need the ingested file to be objectized.

**Note:** Objectization is a resource intensive process. The resource utilization is related to the number of containers that have unified file and object access enabled. The schedule of running the objectization process must be planned carefully. Running it too frequently might impact your protocol node's resource utilization. It is recommended to either schedule it during off business hours, especially if you have a small number of protocol nodes (say 2) with basic resource configuration, or schedule with an interval of 30 minutes or more if you have protocol nodes with adequate resources (where the number of protocol nodes > 2). **It is recommended to set the objectizer service interval to a minimum of 30 minutes or more irrespective of your setup.** If you need to urgently objectize files then you can use the **mmobj file-access** command that allows you to immediately objectize the specified files.

- Set up the objectization interval using the **mmobj config change** as follows.

  ```
  mmobj config change --ccrfile spectrum-scale-objectizer.conf \
  --section DEFAULT --property objectization_interval --value 2400
  ```

  This command sets an interval of 40 minutes between the completion of an objectization cycle and the start of the next cycle.

- Verify that the objectization time interval is changed using the **mmobj config list** as follows.

  ```
  mmobj config list --ccrfile spectrum-scale-objectizer.conf --section DEFAULT
  --property objectization_interval
  ```

## Enabling and disabling QOS

This topic lists the commands to enable and disable QOS.

The periodic scans run by the ibmobjectizer service are resource intensive and might affect the object IO performance. Quality Of Service (QOS) can be set on the ibmobjectizer service depending upon the IO workload and the priority at which the ibmobjectizer service must be run. The usage of resources is limited to the given number so that other high priority workflows and processes can continue with adequate resources, thereby maintaining the performance of the system.

- To enable QOS, type `mmchqos <fs> --enable`.
- Set the **qos_iops_target** parameter in the spectrum-scale-objectizer.conf file. The following example is on one line:

  ```
  mmobj config change --ccrfile spectrum-scale-objectizer.conf --section DEFAULT --property
  qos_iops_target --value 400
  ```

- To disable QOS on ibmobjectizer, set the **qos_iops_target** to 0. The following example is on one line:

  ```
  mmobj config change --ccrfile spectrum-scale-objectizer.conf --section DEFAULT --property
  qos_iops_target --value 0
  ```

## Configuring authentication and setting identity management modes for unified file and object access

You can configure authentication and set the identity management modes for unified file and object access using the following steps.

The identity management modes for unified file and object access are set in the `object-server-sof.conf` file. The default mode is `local_mode`.

**Note:** It is important to understand the identity management modes for unified file and object access and set the mode you want accordingly. Although it is possible to move from one mode to another, some considerations apply in that scenario.

The `unified_mode` identity management mode for unified file and object access is supported only with Active Directory (AD) with UNIX-mapped domains and LDAP authentication configurations. This mode must not be configured with local or user-defined authentication configurations.

**Important:** If you are using `unified_mode`, the authentication for both file and object access must be configured and the authentication schemes must be the same and configured with the same server. If not, the request to create object might fail with user not found error.

Use the following steps on a protocol node to configure authentication and enable `unified_mode`.

1. Determine which authentication scheme best suits your requirements. You can use either LDAP or AD with UNIX-mapped domains.

   **Note:** Because object can be configured with only one AD domain, you need to plan which of the UNIX-mapped AD domains, in case there are trusted domains, is to be configured for object.

2. Configure file access using the **mmuserauth** command as follows.

   ```
   mmuserauth service create --data-access-method file
   --type ad --servers myADserver --idmap-role master
   --netbios-name scale --unixmap-domains 'DOMAIN(5000-20000)'
   ```

3. Configure object access using the **mmuserauth** command as follows.

   ```
   mmuserauth service create --data-access-method object –type ad
   --user-name "cn=Administrator,cn=Users,dc=IBM,dc=local" --password "just4YOU"
   --base-dn "dc=IBM,DC=local" --ks-dns-name c40bbc2xn3 --ks-admin-user admin
   --servers myADserver --user-id-attrib cn --user-name-attrib sAMAccountName
   --user-objectclass organizationalPerson --user-dn "cn=Users,dc=IBM,dc=local"
   --ks-swift-user swift --ks-swift-pwd Passw0rd
   ```

4. Change **id_mgmt** in the `object-server-sof.conf` file using the **mmobj config change** command as follows.

   ```
   mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
   --property id_mgmt --value unified_mode
   ```

5. If object authentication is configured with AD, set **ad_domain** in the `object-server-sof.conf` file.

   ```
   mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
   --property ad_domain --value POLLUX
   ```

   **Note:** Do not specify **ad_domain** with LDAP configurations.

   To find the correct **ad_domain** name, use the following command:

   ```
   /usr/lpp/mmfs/bin/net ads lookup -S {AD_SERVER_NAME | AD_SERVER_IP} -d0
   ```

   For example, in the output of the following command, the value of the **Pre-Win2k Domain** field is the **ad_domain**.

   ```
   /usr/lpp/mmfs/bin/net ads lookup -S 192.196.79.34 -d0

   ...
   Forest:   pollux.com
   Domain:   pollux.com
   Domain Controller: win2k8.pollux.com
   Pre-Win2k Domain: POLLUX
   Pre-Win2k Hostname: WIN2K8
   Server Site Name :  Default-First-Site-Name
   Client Site Name :  Default-First-Site-Name
   ...
   ```

Your unified file and object access enabled fileset is now configured with `unified_mode`.

6. List the currently configured **id_mgmt** mode using the **mmobj config list** command as follows.

```
mmobj config list --ccrfile object-server-sof.conf --section DEFAULT --property id_mgmt
```

**Important:**

1. If the PUT requests fail in **unified_mode**, check if the user name is resolvable on the protocol nodes using the following command:

   ```
   id '<user_name>'
   ```

   If user name in AD is in the domain\user_name format, use the following command:

   ```
   id '<domain\><user_name>'
   ```

2. Ensure that the **ad_domain** parameter is not present in the object-server-sof.conf file when LDAP is configured.
   - To list the object-server-sof.conf file contents, use the following command:

     ```
     mmobj config list --ccrfile object-server-sof.conf
     ```

   - If **ad_domain** is present, remove it as follows:
     a. Copy /etc/swift/object-server-sof.conf to a temporary location, say /tmp.
     b. Modify the temporary file by appending a '-' before the **ad_domain** parameter. This marks that parameter for deletion.
     c. Upload the modified file using the following command:

        ```
        mmobj config change --ccrfile object-server-sof.conf --merge-file /tmp/object-server-sof.conf
        ```

     d. **[Optional]:** Validate that **ad_domain** is removed from the object-server-sof.conf file by listing the file contents.

3. Configuring file authentication with the same scheme as that of object authentication is a mandatory prerequisite before you enable the **unified_mode** identity management mode. In case you configure file authentication later, you must restart swift on the file server for the changes to be effective. You can do this by changing **id_mgmt** to local_mode and then changing it back to unified_mode using the following commands.

   ```
   mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
   --property id_mgmt --value local_mode
   mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
   --property id_mgmt --value unified_mode
   ```

## Creating or using a unified file and object access storage policy

Use the following steps to create or use a unified file and object access storage policy.

1. Create a unified file and object access storage policy using the **mmobj policy create** command. This step also creates a fileset.

   For example:

   ```
   mmobj policy create sof-policy1 --enable-file-access
   ```

   The system displays output similar to the following:

```
[I] Getting latest configuration from ccr
[I] Creating fileset /dev/gpfs0:obj_sof-policy1
[I] Creating new unique index and build the object rings
[I] Updating the configuration
[I] Uploading the changed configuration
```

2. List the available storage policies using the **mmobj policy list** command and determine which policies are for unified file and object access by viewing the **Functions** column of the output.

   For example:

   ```
   mmobj policy list --verbose
   ```

   The system displays output similar to the following:

```
Index       Name          Deprecated  Fileset          Fileset Path                      Functions               Function Details
-------------------------------------------------------------------------------------------------------------------------------
0           SwiftDefault              object_fileset   /ibm/cesSharedRoot/object_fileset
11751509160 sof-policy1               obj_sof-policy1  /ibm/cesSharedRoot/obj_sof-policy1  file-and-object-access  regions="1"
11751509230 mysofpolicy               obj_mysofpolicy  /ibm/cesSharedRoot/obj_mysofpolicy  file-and-object-access  regions="1"
11751510260 Test19                    obj_Test19       /ibm/cesSharedRoot/obj_Test19                               regions="1"
```

3. Start using one of these storage policies to create data in a unified file and object access environment.

For more information, see the following:

- "Associating containers with unified file and object access storage policy"
- "Creating exports on container associated with unified file and object access storage policy"

For information on mapping of storage policy and filesets, see "Mapping of storage policies to filesets" on page 191.

You must create export at the container level. From NFS or SMB, if you create a peer container, base containers created from NFS and SMB cannot be multiprotocol.

## Associating containers with unified file and object access storage policy

Use the following steps to associate a container with a unified file and object access storage policy.

1. Associate a container with a unified file and object access storage policy using the following command.

   ```
   swift post container1  --os-auth-url http://specscaleswift.example.com:35357/v3
   --os-project-name admin --os-project-domain-name Default --os-username admin
   --os-user-domain-name Default --os-password Passw0rd --auth-version 3
   --header "X-Storage-Policy: sof-policy1"
   ```

   In this **swift post** example, the storage policy is specified with the customized header `X-Storage-Policy` using the `--header` option.

2. Upload an object in the container associated with the unified file and object access storage policy using the following command.

   ```
   swift upload container1 imageA.JPG --os-auth-url http://specscaleswift.example.com:35357/v3
   --os-project-name admin --os-project-domain-name Default --os-username admin
   --os-user-domain-name Default --os-password Passw0rd --auth-version 3
   object1
   ```

**Note:** The steps performed using **swift** commands can also be done using **curl**. For more information, see "curl commands for unified file and object access related user tasks" on page 216.

## Creating exports on container associated with unified file and object access storage policy

Use the following steps to create an NFS or SMB export on the directory that maps to the container associated with the unified file and object access storage policy.

Create an SMB or NFS export on the directory that maps to the container associated with the unified file and object access storage policy.

1. Create the NFS export as follows:

   ```
   mmnfs export add "/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/
   AUTH_76347638472849832 3747/cont"
   ```

2. Create the SMB share as follows:

   ```
   mmsmb export add smbexport "/ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/
   AUTH_76347638472849832 3747/cont"
   ```

**Note:**

- It is strongly recommended that you create file exports on or below the container path level and not above it. Creating file exports above the container path level might lead to deletion of the unified file and object access enabled containers which is undesirable.
- When using POSIX interface, it is strongly recommended to only allow access of data to POSIX users from on or below the container path. Accidental deletion of container or data above might lead to inconsistent state of the system.

## Enabling object access for selected files

Use the following steps to objectize files under all the containers associated with the unified file and object access storage policy under an account

In a unified file and object access environment, if you want to access files created from file interfaces such as POSIX, NFS, or CIFS through object interfaces such as curl or swift, you need to make these files available for the object interface. For making these files available for the object interface, the **ibmobjectizer** service, once activated, runs periodically and makes newly created files available for the object interface. You can also use the **mmobj file-access** command to selectively enable files for access through the object interface.

The purpose of this command is to make certain files available to object sooner (or immediately) than when the objectizer would have made them available. This command does not ensure synchronization between file and object data. Therefore, files deleted are not immediately reflected in the object interface. Complete synchronization is done by the **ibmobjectizer** service eventually.

In unified file and object access enabled filesets, files can be accessed from the object interface if you know the entire URI (including keystone account ID, device etc.) to access that file without the need for them to be objectized either using the **ibmobjectizer** service or the **mmobj file-access** command.

**Note:** The **mmobj file-access** command does not enable or disable the unified file and object access feature. It is only used to objectize files (that is enable object access for files) immediately when initiated by the administrator. Disabling object access for files is not supported.

- To objectize files under all the containers associated with the unified file and object access storage policy under an account, use the `mmobj file-access` command as follows:

  ```
  mmobj file-access --storage-policy sof_policy --account-name admin
  ```

  The system displays output similar to the following:

  ```
  Loading objectization configuration from CCR
  Fetching storage policy details
  Performing objectization
  Objectization complete
  ```

  This command objectizes all containers in the account `admin` and enables them for access through the object interface.
- To objectize files under a container, use the `mmobj file-access` command as follows:

  ```
  mmobj file-access --storage-policy sof_policy --account-name admin --container-name container1
  ```

  This command objectizes all files in `container1` and enables them for access through the object interface.
- To objectize a file while specifying a storage policy, use the **mmobj file-access** command with the `--storage-policy` option as follows:

  ```
  mmobj file-access --storage-policy sof_policy --account-name admin \
  --container-name container1 --object-name file1.txt
  ```

  This command objectizes `file1.txt` in `container1` and enables it for access through the object interface.
- To objectize a file, use the **mmobj file-access** command as follows:

  ```
  mmobj file-access --object-path \
  /ibm/gpfs0/obj_sofpolicy1/s69931509221z1device1/AUTH_763476384728498323747/cont/file1.txt
  ```

  This command objectizes `file1.txt` at location `/ibm/cesSharedRoot/fileset1/Auth_12345/container1/` and enables it for access through the object interface.

For more information about the **mmobj file-access** command, see *mmobj command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Example scenario - administering unified file and object access

The following example describes an end-to-end scenario of administering and using unified file and object access.

Before you can use the following steps, IBM Spectrum Scale for object storage must be installed.

This example provides a quick reference of steps performed for unified file and object access. For detailed information about these steps, see "Administering unified file and object access" on page 205.

1. Enable the file-access object capability as follows.

   ```
   mmobj config change --ccrfile spectrum-scale-object.conf \
   --section capabilities --property file-access-enabled --value true
   ```

2. [Optional] Change the objectizer service interval as follows.

   ```
   mmobj config change --ccrfile spectrum-scale-objectizer.conf \
   --section DEFAULT --property objectization_interval --value 600
   ```

3. [Optional] Change the identity management mode to `unified_mode` as follows.

   ```
   mmobj config change --ccrfile object-server-sof.conf \
   --section DEFAULT --property id_mgmt --value unified_mode
   ```

4. [Optional] Set the `ad_domain` parameter as follows.

   ```
   mmobj config change --ccrfile object-server-sof.conf \
   --section DEFAULT --property ad_domain --value ADDOMAINX
   ```

5. Create a unified file and object access storage policy as follows.

   ```
   mmobj policy create SwiftOnFileFS --enable-file-access
   ```

   The system displays output similar to the following:

   ```
   [I] Getting latest configuration from ccr
   [I] Creating fileset /dev/gpfs0:obj_SwiftOnFileFS
   [I] Creating new unique index and building the object rings
   [I] Updating the configuration
   [I] Uploading the changed configuration
   ```

   This command also creates a unified file and object access enabled fileset.

6. Create a base container with a unified file and object access storage policy as follows.

   ```
   swift post unified_access -H "X-Storage-Policy: SwiftOnFileFS"
   ```

7. Store the path created for the container by finding it in the newly created fileset as follows.

   ```
   export FILE_EXPORT_PATH=`find  /ibm/gpfs0/obj_SwiftOnFileFS/
    -name "unified_access"`

   # echo $FILE_EXPORT_PATH
   /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
   AUTH_09271462d54b472c82adecff17217586/unified_access
   ```

8. Create an SMB share on the path as follows.

   ```
   mmsmb export add unified_access $FILE_EXPORT_PATH
   ```

   The system displays output similar to the following:

   ```
   mmsmb export add: The SMB export was created successfully
   ```

9. Create an NFS export on the path.

   ```
    mmnfs export add  $FILE_EXPORT_PATH --client \
   "*(Access_Type=RW,Squash=no_root_squash,SecType=sys)"
   ```

   The system displays output similar to the following:

   ```
   192.0.2.2:  Redirecting to /bin/systemctl stop  nfs-ganesha.service
   192.0.2.3:  Redirecting to /bin/systemctl stop  nfs-ganesha.service
   192.0.2.2:  Redirecting to /bin/systemctl start  nfs-ganesha.service
   192.0.2.3:  Redirecting to /bin/systemctl start  nfs-ganesha.service
   NFS Configuration successfully changed. NFS server restarted on all NFS nodes.
   ```

   **Note:** If this is the first NFS export added to the configuration, the NFS service will be restarted on the CES nodes where the NFS server is running. Otherwise, no NFS restart is required when adding an NFS export.

10. Check the NFS and SMB shares.

    ```
    mmnfs export list
    ```

    The system displays an output similar to the following:

```
Path Delegations Clients
------------------------------------------------------------------------------
/ibm/gpfs0/obj_SwiftOnFileFS/
s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access none *
mmsmb export list

export          path                                    guest ok    smb encrypt
unified_access   /ibm/gpfs0/obj_SwiftOnFileFS/
s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access     no          auto

Information:
The following options are not displayed because they do not contain a value:
  "browseable"
```

11. Access this export with NFS or SMB clients and create a sample directory and a file. For example: `DirCreatedFromGPFS/File1.txt` and `DirCreatedFromSMB/File2.txt`

    You can view the association of ownership when data is created from the SMB interface as follows.

```
ls -l /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB
total 0
-rwxr--r--. 1 ADDOMAINX\administrator
ADDOMAINX\domain users 20 Oct 21 18:09 File2.txt

mmgetacl /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB
#NFSv4 ACL
#owner:ADDOMAINX\administrator
#group:ADDOMAINX\domain users
special:owner@:rwxc:allow
 (X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (X)SYNCHRONIZE
 (X)READ_ACL  (X)READ_ATTR  (X)READ_NAMED
 (-)DELETE    (X)DELETE_CHILD (X)CHOWN
 (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (X)WRITE_NAMED

special:group@:r-x-:allow
 (X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE
 (X)READ_ACL  (X)READ_ATTR  (X)READ_NAMED
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN
 (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

special:everyone@:r-x-:allow
 (X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE
 (X)READ_ACL  (X)READ_ATTR  (X)READ_NAMED
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN
 (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

    You can view the container and the file created from the REST interface and retention of ownership in the PUT operation as follows.

```
ls -l /ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/
AUTH_09271462d54b472c82adecff17217586/unified_access/DirCreatedFromSMB/File2.txt

-rwxr-xr-x. 1 ADDOMAINX\administrator ADDOMAINX\domain users 520038360 Nov 3 11:47
/ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/AUTH_09271462d54b472c82adecff17217586/
DirCreatedFromSMB/unified_access/File2.txt
```

12. Objectize that file immediately by using the following command or wait for the objectization cycle to complete.

```
mmobj file-access --object-path \
/ibm/gpfs0/obj_SwiftOnFileFS/s10041510210z1device1/AUTH_09271462d54b472c82adecff17217586
/unified_access/File2.txt
```

13. Download that object using the Swift client which is configured with all variables as follows.

```
swift download unified_access/File2.txt
```

14. List the contents of the container using the Swift client which is configured with all variables as follows.

```
swift list unified_access
```

    The system displays output similar to the following:

```
DirCreatedFromGPFS/File1.txt
DirCreatedFromSMB/File2.txt
```

**Note:** The steps performed using **swift** commands can also be done using **curl**. For more information, see "curl commands for unified file and object access related user tasks" on page 216.

## In-place analytics using unified file and object access

Use the following information to leverage in-place object data analytics using unified file and object access.

Unified file and object access is one of the key features of IBM Spectrum Scale for object storage that enables direct object access as files from the traditional file access such as POSIX, NFS or SMB and vice versa. Using object storage policies for containers you can have object ingested in IBM Spectrum Scale for object storage be accessed as files as well as allow files ingested using file protocols available for object access. This feature enables data analytics of object data hosted on IBM Spectrum Scale, where you can leverage in-place object data analytics. IBM Spectrum Scale supports Hadoop connectors using which you can run analytics on the object data which is accessible from the file interface and generates in-place results which are directly accessible from the object interface. This prevents any movement of data across object interfaces and thus proves to be a suitable platform for object storage as well as integrated in-place analytics for the data hosted by it.

The following diagram shows an IBM Spectrum Scale object store with unified file and object access. The object data is available as file on the same fileset. IBM Spectrum Scale Hadoop connectors allow the data to be directly leveraged for analytics.

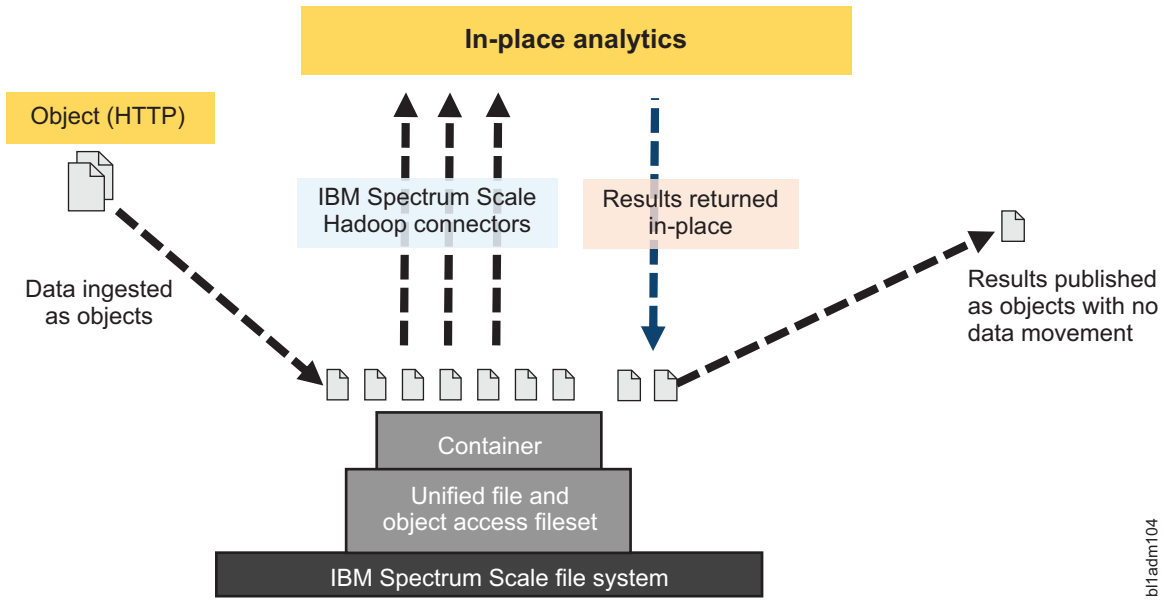



*Figure 5. In-place analytics with unified file and object access*

## Limitations of unified file and object access

The following limitations apply for unified file and object access in IBM Spectrum Scale.

- Existing file data cannot be enabled for object access. The base container must be created from the object interface in the fileset being used for the unified access storage policy and then only the data added after that is enabled for object access.
- Concurrent access to the same object or file from file and object interface at the same time will lead to an undefined state. There are a variety of ways to prevent conflicts. For example:
  1. You can have your workflow enforce this.

2. You can explicitly enforce read-only access for some periods. With NFS and SMB, it can be done in the export definition. With POSIX, it can be done using ACLs.

- Files or directories created at the base container level cannot be enabled for object access. Only the files created under the container are enabled for object access.
- Multi-region object deployment cannot be used with unified file and object access.
- Object versioning is not supported with unified file and object access.
- Special files such as device files and pipes, and soft links can exist in the object container directory, but they are not visible from the object interface.
- AFM-based Async DR is supported with unified file and object access. No other active file management (AFM) modes are supported with unified file and object access.
- Containers must be deleted from the object interface. Container directories deleted from the file interface continue to show up in the container listing, until the container is deleted from the object interface.
- GPFS quota and Swift container and account quota are mutually exclusive in Release 4.2 and later. The user quota assigned to a user or a group in GPFS does not relate to the container quota defined in the object interface.
- Swift large object support (dynamic large object and static large object) is not available with unified file and object-enabled containers. S3 multipart uploads are also not supported with unified file and object-enabled containers.
- GPFS immutability is not supported with unified file and object access.
- Only object metadata can be viewed and modified from the object interface. Extended attributes defined from the file interface cannot be viewed from the object interface.
- Empty directories created from the file interface within a container are not objectized and they are not listed in the container listing.
- Files or directories with "::" or newline characters in their names are not supported and these files or data residing in these containers are not objectized.
- Change of authentication scheme of file or object could directly impact access to existing file or object data. Therefore, change of authentication is not supported as it results in loss of access for the users to the existing data on the system.
- Object ETag is inaccurate in the following scenarios:
  - Whenever an object is modified from the file interface. In this case, performing a conditional request using 'If-Match' or 'If-None-Match' headers returns incorrect results.
  - ETag for files on an explicit GET request when the size of the object has changed (increased or decreased).
  - ETag for files on an explicit GET request when the content of the object has changed but the size has remained the same.
  - If the **user.swift.metadata** extended attribute is explicitly deleted form the file interface, ETag is not present because of which the headers do not return correct results. Users need to wait for at least one cycle of objectization or they need to explicitly objectize that file to use the ETag conditional request feature.

  **Note:** An incorrect ETag is corrected when a GET or HEAD request is performed on the object.
- The IBM Spectrum Scale ILM policy rules work with file-extended attributes, and rules can be easily created based on extended attributes and their values. However, these rules do not work directly over Swift user-defined metadata. All of Swift user-defined metadata is stored in a single extended attribute in the IBM Spectrum Scale filesystem. To create ILM rules, the format and sequence in which the attributes are stored must be noted. Rules can then be created by constructing wildcard-based filters.
- Object versioning is not supported for unified file and object containers.
- To enable object access for the existing filesets, SELinux must be in the Permissive or Disabled mode.

- Files such as device files, pipes, and soft links can exist in the object container directory, but they cannot be viewed from the object interface for enabling object access.
- If the `--update-listing yes` option of the **mmobj file-access link-fileset** command is used, then the `source-fsetpath` must be the exact fileset junction path and the fileset must be derived from the object file system.
- The conditional client, such as swift and curl, features such as `If-Match` and `If-None-Match` that performs ETag comparison does not work for the existing data enabled for object access by using by `--update-listing no` of the **mmobj file-access link-fileset** command . If the `--update-listing yes` option is used, the feature can be used after the objectizer service interval.

## Constraints applicable to unified file and object access

The following constraints are applicable while creating and accessing objects and containers for unified file and object access:

- The name of the container must not exceed 255 characters.
- The name of the object must not exceed 214 characters.
- The path name of the object must not include successive forward slashes.
- The name of the container and the object must not be a single period (.) or a double period (..). However, a single period or a double period can be part of the name of the container and the object.

The system returns 400 Bad Request when the above constraints are not met.

The swift constraints listed in the following table are also applicable to unified file and object access.

*Table 21. Configuration options for [swift-constraints] in swift.conf*

| Option | Limit |
|---|---|
| MAX_FILE_SIZE | 5497558138880 (5 TiB) |
| MAX_META_NAME_LENGTH | 128 |
| MAX_META_VALUE_LENGTH | 256 |
| MAX_META_COUNT | 90 |
| MAX_META_OVERALL_SIZE | 4096 |
| MAX_HEADER_SIZE | 8192 |
| CONTAINER_LISTING_LIMIT | 10000 |
| ACCOUNT_LISTING_LIMIT | 10000 |
| MAX_ACCOUNT_NAME_LENGTH | 256 |
| VALID_API_VERSIONS | ["v1", "v1.0"] |
| EXTRA_HEADER_COUNT | 0 |

**Note:** These values can be changed by using **mmobj config change** command for the swift.conf file in swift-constraints section.

## Data ingestion examples

Use the following example steps for data ingestion in the following scenarios.

- Data ingestion through object interface and access through file interface
- Data ingestion through file interface and access through object interface
- Data ingestion and access through object and file interfaces concurrently

1. **Standard REST client step:** Get proper authentication token from the Authentication URL using proper credentials to authorize on further requests.

2. **Standard REST client step:** Using token obtained in the previous step perform PUT, POST, DELETE, COPY (object only), HEAD operations for objects under container created with unified file and object access storage policy.
3. **Standard file client step:** Mount SMB or NFS exports on respective NFS or SMB clients with regular mount commands or interface available with file clients. For example:

```
mount -t cifs -o username=STORAGE5TEST\\fileuser1,password=Passw0rd5,vers=3.0 //192.0.2.4/unified_access /mnt/unified_access
```

**Data ingestion using curl**

In the following data ingestion example steps performed using **curl**, this setup is assumed:

- User: "fileuser"
- Password: "Password6"
- Account: "admin"
- Host: specscaleswift.example.com

1. Obtain the auth token using the following command:

```
curl -s -i -H "Content-Type: application/json"
-d '{ "auth": {"identity": {"methods": ["password"],"password": {"user": {"name": "fileuser","domain":
{ "name": "Default" },"password": "Passw0rd6"}}},"scope": {"project": {"name": "admin","domain": { "name": "Default" }}}}}'
http://specscaleswift.example.com:35357/v3/auth/tokens
```

   The auth token obtained in this step must be stored in the *$AUTH_TOKEN* variable.

2. Obtain the project list using the following command:

```
curl -s -H "X-Auth-Token:  $AUTH_TOKEN" http://specscaleswift.example.com:35357/v3/projects
```

   The project ID obtained in this step must be stored in the *$AUTH_ID* variable.

3. Perform a PUT operation using the following command:

```
curl -i -s -X PUT --data @/tmp/file.txt -H "X-Auth-Token:
$AUTH_TOKEN" "http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt"
```

   This command uploads the `/tmp/file.txt` file.

4. Set up the metadata age of the uploaded object using the following command:

```
curl -i -s -X POST -H "X-Auth-Token: $AUTH_TOKEN" -H
X-Container-Meta-Age:21 http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt
```

5. Read the metadata using the following command:

```
curl -i -s --head -H  "X-Auth-Token: $AUTH_TOKEN"
http://specscaleswift.example.com:8080/v1/AUTH_$AUTH_ID/RootLevelContainer/TestObj.txt
```

## curl commands for unified file and object access related user tasks

Use the following curl commands to perform user tasks related to unified file and object access.

For the following commands, it is assumed that:

- A token is generated and it is exported as an environment variable *AUTH_TOKEN*.
- A swift endpoint URL for the project (tenant) for which token has been generated.
- A unified file and object access storage policy named `SwiftOnFileFS` is already created.

1. Create a container named `unified_access` with unified file and object access storage policy using **curl** as follows.

   ```
   curl -v -i -H "X-Auth-Token: $AUTH_TOKEN"
   -X PUT http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df/unified_access/
   -H "X-Storage-Policy: SwiftOnFileFS"
   ```

   In this command, `http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df/` is the endpoint URL for a project (tenant) using which a container `unified_access` is created with `SwiftOnFileFS` as the storage policy.

2. Upload an object in the container associated with the unified file and object access storage policy using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $AUTH_TOKEN"
-X PUT http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/object1
--data-binary @imageA.jpg
```

3. Download object residing in the unified file and object access container using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $AUTH_TOKEN"
-X GET http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/samplefile.txt
```

4. List the contents of the unified file and object access container using **curl** as follows.

```
curl -v -i -H "X-Auth-Token: $AUTH_TOKEN"
-X GET http://specscaleswift.example.com:8080/v1/AUTH_cd1a29013b6842939a959dbda95835df
/unified_access/
```

# Configuration files for IBM Spectrum Scale for object storage

Use the following information to manage options in configuration files that are used for IBM Spectrum Scale for object storage including the unified file and object access feature. These configuration files are located in the `/etc/swift` directory.

For information on changing an option in a configuration file, see "Changing options in configuration files" on page 220.

## object-server-sof.conf

- Contains identity management modes for unified file and object access (**id_mgmt**)
- Contains AD domain name (**ad_domain**) if AD is configured

*Table 22. Configurable options for [DEFAULT] in object-server-sof.conf*

| Configuration option = Default value | Description |
|---|---|
| **id_mgmt** = local_mode | Defines the object server behavior while assigning user or group ownership to newly created objects, when those are accessed using the file interface. The allowed values are `local_mode` and `unified_mode`<br><br>With `local_mode`, the new objects are owned by the `swift` user. In `unified_mode`, the identity of the user making the PUT request is fetched from the configured directory server. |
| **ad_domain** | When using Active Directory (AD), defines the AD domain from which the user identity should be fetched when object server is operating in the `unified_mode` identity management mode.<br>**Note:** When you clean up object authentication, you must manually remove this entry. For more information, see "Configuring authentication and setting identity management modes for unified file and object access" on page 206. |
| **tempfile_prefix** = .ibmtmp_ | The prefix to be used for the temporary file being created when a file being uploaded. |
| **disable_fallocate** = true | Overrides the default swift fallocate behavior, and relies on the GPFS fallocate features, excludes 'fast fail' checks. |
| **disk_chunk_size** = 65536 | The size of chunks to read/write to disk (needs be equal to the file system block size). |
| **network_chunk_size** = 65536 | The size of chunks to read/write over the network (needs be equal to the file system block size). |
| **log_statsd_host** = localhost | If not set, the StatsD feature is disabled. |

*Table 22. Configurable options for [DEFAULT] in object-server-sof.conf (continued)*

| Configuration option = Default value | Description |
|---|---|
| **log_statsd_port** = 8125 | The port number for the StatsD server. |
| **log_statsd_default_sample_rate** = 1.0 | Defines the probability of sending a sample for any given event or timing measurement. |
| **log_statsd_sample_rate_factor** = 1.0 | Not recommended to set this to a value less than 1.0. If the frequency of logging is too high, tune the **log_statsd_default_sample_rate** instead. |
| **log_statsd_metric_prefix** = | The prefix that is added to every metric sent to the StatsD server. |
| **retain_acl** = yes | Specifies whether or not to copy the ACL from an existing object. Allowed values are yes or no. |
| **retain_winattr** = yes | Specifies whether or not to copy the Windows attributes from an existing object. Allowed values are yes or no. |
| **retain_xattr** = yes | Specifies whether or not to copy the extended attributes for the user namespace from an existing object. Allowed values are yes or no. |
| **retain_owner** = yes | Specifies whether or not to copy the UID/GID owners from an existing object. Allowed values are yes or no. |

**Note:** Files with the `.ibmtmp` prefix or the one configured in the `object-server-sof.conf` configuration file are not objectized.

When you set the `retain_*` options to `yes`, the following attributes are retained:

- The extended attributes in the `user` namespace except for the **user.swift.metadata** key which contains swift metadata and it is expected to be new.
- Windows attributes

When you set the `retain_*` options to `yes`, the following attributes are not retained:

- Extended attributes in `system`, `security`, and `trusted` namespaces.

  **Note:** These attributes are not retained in an object's copy object operation also.

Retaining ACLs, Windows attributes, file extended attributes, and ownership, when an object is PUT over an existing object in unified file and object access enabled containers depends on your specific use case and your discretion. For example, if you are using object and file access to refer to the same data content in such a way that the object protocol might completely replace the data content in such that it might be completely new content from the file interface as well, then you might choose to not retain the existing file ACL and extended attributes. For such a use case, you might change the default values to not to retain the file ACLs, extended attributes, and ownership.

**Note:** If you are unsure about whether to retain these attributes or not, you might want to use the default values of retaining ACLs, Windows attributes, file extended attributes, and ownership. The default values in this case are more aligned with the expected behavior in a multiprotocol setup.

## spectrum-scale-object.conf

- Contains cluster or fileset configuration information
- Unique to a site

*Table 23. Configurable options for [capabilities] in spectrum-scale-object.conf*

| Configuration option = Default value | Description |
|---|---|
| **file-access-enabled** = false | The state for the **file-access** capability. It can be either true or false. |
| **multi-region-enabled** = true | The state for the **multi-region** capability. This option cannot be changed. |
| **s3-enabled** = true | The state for the **s3** capability. This option cannot be changed. |

## spectrum-scale-objectizer.conf

•

Contains the **ibmobjectizer** service configuration information

*Table 24. Configuration options for [DEFAULT] in spectrum-scale-objectizer.conf.*

| Configuration option = Default value | Description |
|---|---|
| **objectization_tmp_dir** | The temporary directory to be used by ibmobjectizer. This must be a path on any GPFS file system. The default value is autofilled with the path of the base file system for object. |
| **objectization_threads** = 24 | The maximum number of threads that ibmobjectizer will spawn on a node. |
| **batch_size** = 100 | The maximum number of files that ibmobjectizer will process in a thread. |
| **objectization_interval** = 1800 | The time interval, in seconds, between the completion of an objectization cycle and the beginning of the next objectization cycle. |
| **conection_timeout** = 25 | The connection time out for an account, container, or object server request from ibmobjectizer, |
| **response_timeout** = 25 | The response time out for an account, container, object server request from ibmobjectizer |
| **qos_iops_target** = 0 | The value assigned to ibmobjectizer to limit its resource usage. Value is given in IOPS unit. For example- 100, 400, 0.<br><br>0 means infinite. |

*Table 25. Configuration options for [IBMOBJECTIZER-LOGGER] in spectrum-scale-objectizer.conf.*

| Configuration option = Default value | Description |
|---|---|
| **log_level** = INFO | The logging level. Allowed value is one of the following: INFO, DEBUG, WARN, ERROR |

## object-server.conf file

•

Used to set swift timeout values on the lock_path calls to handle GPFS delays better

*Table 26. Configuration options for object-server.conf*

| Configuration option = Default value | Description |
|---|---|
| **partition_lock_timeout** = 10 | The time-out value while the object server tries to acquire a lock on the partition path during object create. update, and delete processes. The default value is 10. |

### Changing options in configuration files

You can use the **mmobj config change** command to change the values of the options in the configuration files. For example:

- Change the value of an option in the [DEFAULT] section of the `object-server-sof.conf` file as follows:

```
mmobj config change --ccrfile object-server-sof.conf --section DEFAULT
--property OPTIONNAME --value NEWVALUE
```

- Change the value of an option in the [IBMOBJECTIZER-LOGGER] section of the `spectrum-scale-objectizer.conf` file as follows:

```
mmobj config change --ccrfile spectrum-scale-objectizer.conf --section IBMOBJECTIZER-LOGGER
--property OPTIONNAME --value NEWVALUE
```

**Note:** Only some options are configurable. If an option cannot be changed, it is mentioned in the respective option description.

**Attention:** When a configuration file is changed using these commands, it takes several seconds for the changes to be synced across the whole cluster depending on the size of the cluster. Therefore, when executing multiple commands to change configuration files, you must plan for an adequate time interval between the execution of these commands.

## Backing up and restoring object storage

Snapshots are a good way to protect data from various errors and failures. Moving them to a separate backup storage system can provide better protection against catastrophic failures of the entire storage system and might even allow the data to be stored at a lower cost. This section describes the manual steps that are needed to back up and restore the object storage and its configuration information.

In the examples, the steps to back up the Keystone configuration files and database are not given. That is the user's responsibility. You can use OpenStack backup procedures for this task. For more information on OpenStack backup procedures, see Chapter 14. Backup and Recovery.

**Note:**

- The same version of the IBM Spectrum Protect backup-archive client must be installed on all of the nodes that are running the **mmbackup** command.
- For more information on IBM Spectrum Protect requirements for the **mmbackup** command, see *IBM Spectrum Scale requirements* in *IBM Spectrum Scale: Administration Guide*.

## Backing up the object storage

All IBM Spectrum Scale Object Nodes and IBM Spectrum Protect client nodes must be available with the object file system mounted on each node when the backup is being created. The IBM Spectrum Protect server must also be available.

Store all relevant cluster and file system configuration data in a safe location outside your GPFS cluster environment. This data is essential to restoring your object storage quickly, so you might want to store it in a site in a different geographical location for added safety.

Follow these steps to back up the object storage manually:

**Note:** The sample file system used throughout this procedure is called **smallfs**. Replace this value with your file system name wherever necessary.

1. Back up the cluster configuration information.

The cluster configuration must be backed up by the administrator. The following cluster configuration information is necessary for the backup:

- IP addresses
- Node names
- Roles
- Quorum and server roles
- Cluster-wide configuration settings from mmchconfig
- Cluster manager node roles
- Remote shell configuration
- Mutual ssh and rsh authentication setup
- Cluster UID

**Note:** Complete configuration information can be found in the `mmsdrfs` file.

2. Preserve disk configuration information.

   Disk configuration must also be preserved to recover a file system. The basic disk configuration information needed for a backup intended for disaster recovery is:

   - The number of disk volumes that were previously available
   - The sizes of those volumes

   To recover from a complete file system loss, at least as much disk space as was previously available is needed for restoration. It is only possible to restore the image of a file system onto replacement disks if the disk volumes available are of similar enough sizes to the originals that all data can be restored to the new disks. The following disk configuration information is necessary for the recovery:

   - Disk device names
   - Disk device sizes
   - The number of disk volumes
   - NSD server configuration
   - Disk RAID configurations
   - Failure group designations
   - The `mmsdrfs` file contents

3. Back up the GPFS™ file system configuration information.
   In addition to the disks, the file system built on those disks has the following configuration information that can be captured using the **mmbackupconfig** command:

   - Block size
   - Replication factors
   - Number and size of disks
   - Storage pool layout
   - Filesets and junction points
   - Policy rules
   - Quota information
   - Other file system attributes

   The file system configuration information can be backed up into a single file using a command similar to the following:

   ```
   mmbackupconfig smallfs -o /tmp/smallfs.bkpcfg.out925
   ```

4. Save the following IBM Spectrum Protect configuration files for each IBM Spectrum Protect client node in the same safe location outside of your GPFS cluster.

**/etc/adsm/TSM.PWD**
Contains the client password that is needed to access IBM Spectrum Protect. This file is present only when the IBM Spectrum Protect server setting of authentication is set to `on`.

**/opt/tivoli/tsm/client/ba/bin/dsm.sys and**
**/opt/tivoli/tsm/client/ba/bin/dsm.opt**
Contains the IBM Spectrum Protect client configuration files.

5. Issue the **mmcesdr primary backup** command to save the Swift configuration files for all protocol nodes.

   This command stores all of the Swift configuration data in an independent fileset created on the object storage file system. This must be done before creating the global snapshot to preserve and back up the configuration data.

6. Back up the object storage content to an IBM Spectrum Protect server by running the **mmbackup** command:
   a. Create a global snapshot by running the following command:
      **mmcrsnapshot <file system device> <snapshot name>**.
      For example, create a snapshot that is named `objects_globalsnap1` by running the following command:
      **mmcrsnapshot smallfs objects_globalsnap1**
   b. Create global and local work directories by running the following commands:
      **mkdir -p /smallfs0/.es/mmbackupglobal**
      **mkdir -p /smallfs0/.es/mmbackuplocal**
   c. Issue the following command to start the snapshot-based backup:
      **mmbackup <file system device> -t incremental -N <TSM client nodes> \ -g <global work directory> \ -s <local work directory> \-S <global snapshot name> --tsm-servers <tsm server> --noquote**
      The \ indicates the line wrap.
      For example:

```
mmbackup smallfs -t incremental -N node1,node2 \
-g /smallfs0/.es/mmbackupglobal \
-s /smallfs0/.es/mmbackuplocal \
-S objects_globalsnap1 -—tsm-servers tsm1 --noquote
```

      where

      **-N** Specifies the nodes that are involved in the backup process. These nodes must be configured for the IBM Spectrum Protect server that is being used.

      **-S** Specifies the global snapshot name to be used for the backup.

      **--tsm-servers**
      Specifies which IBM Spectrum Protect server is used as the backup target, as specified in theIBM Spectrum Protect client configuration dsm.sys file.

      There are several other parameters available for the **mmbackup** command that influence the backup process, and the speed with which its handles the system load. For example, you can increase the number of backup threads per node by using the **-m** parameter. For the full list of parameters available, see the *mmbackup command* in the *IBM Spectrum Scale: Command and Programming Reference*.
   d. Issue the following command to remove the snapshot that was created in step 6a:
      **mmdelsnapshot <file system device> <snapshot name>**
      For example:
      `mmdelsnapshot smallfs objects_globalsnap1`

## Restoring the object storage

You must meet the following prerequisites before beginning the recovery procedure:

1. Restore the GPFS cluster with the same node names that were used during the backup procedure
2. Restore your OpenStack Keystone server and make sure that it is operational.
3. Install Swift software on all IBM Spectrum Scale object nodes.
4. Install the IBM Spectrum Protect backup-archive client software on the IBM Spectrum Scale object nodes that were clients previously.

   **Note:** All IBM Spectrum Scale object nodes and IBM Spectrum Protect client nodes must be available when the object storage configuration and contents are being restored.

After you perform the prerequisite procedures, you can begin the recovery procedure.

**Note:** The sample file system that is used throughout this procedure is called **smallfs**. Replace this value with your file system name wherever necessary.

1. Retrieve the base file system configuration information.

   Use the **mmrestoreconfig** command to generate a configuration file that contains the details of the former file system. For example:

   ```
   mmrestoreconfig smallfs -i /tmp/smallfs.bkpcfg.out925 -F
   smallfsQueryResultFile
   ```

2. Re-create the NSDs if they are missing.

   Using the output file that is generated in the previous step as a guide, the administrator might need to re-create NSD devices for use with the restored file system. In the output file, the NSD configuration section contains the NSD information. For example:

   ```
   ######## NSD configuration ##################
   ## Disk descriptor format for the mmcrnsd command.
   ## Please edit the disk and desired name fields to match
   ## your current hardware settings.
   ##
   ## The user then can uncomment the descriptor lines and
   ## use this file as input to the -F option.
   #
   # %nsd:
   #   device=DiskName
   #   nsd=nsd8
   #   usage=dataAndMetadata
   #   failureGroup=-1
   #   pool=system
   #
   ```

   If changes are needed, edit the file in a text editor and follow the included instructions to use it as input for the **mmcrnsd** command, then issue the following command:

   ```
   mmcrnsd -F StanzaFile
   ```

3. Re-create the base file system.

   The administrator must re-create the initial file system. The output query file created in step 1 can be used as a guide. The following example shows the section of this file that is needed when re-creating the file system:

   ```
   ######### File system configuration #############
   ## The user can use the predefined options/option values
   ## when recreating the filesystem.  The option values
   ## represent values from the backed up filesystem.
   #
   # mmcrfs FS_NAME NSD_DISKS -j cluster -k posix -Q yes -L 4194304 --disable-fastea
     -T /smallfs -A no --inode-limit 278016#
   ```

4. Restore the essential file system configuration.

The essential file system configuration can be restored to the file system that was created in the previous step by using the **mmrestoreconfig** command. For example:

```
mmrestoreconfig smallfs -i /tmp/smallfs.bkpcfg.out925
```

5. Issue the following command to mount the object file system on all nodes:
   **mmmount <file system device> -a**
   For example, mount the file system with the following command:
   ```
   mmmount smallfs -a
   ```
6. Restore the configuration of the IBM Spectrum Protect client nodes by copying the saved configuration files from their saved location to each IBM Spectrum Protect client node.
   a. The IBM Spectrum Protect client config files `dsm.opt` and `dsm.sys` must be restored to `/opt/tivoli/tsm/client/ba/bin/`.
   b. If the IBM Spectrum Protect client password file, `TSM.PWD`, is saved during the backup procedure, it must be restored to `/etc/adsm/`.
   c. Issue the following command to verify that each IBM Spectrum Protect client node can communicate with the IBM Spectrum Protect server without prompting for a password: **dsmc q sess**
7. Restore the object storage data from the IBM Spectrum Protect server. This also restores the object storage configuration data stored within the filesystem.
   a. Issue the **dsmc restore** command as shown to start a no-query restore on a IBM Spectrum Protect client node.
      ```
      dsmc restore <GPFS Object path> -subdir=yes -disablenqr=no \
      -servername=<tsm server> -errorlogname=<error log path>.
      ```
      For example:
      ```
      dsmc restore /smallfs/ -disablenqr=no \
      -servername=tsm1 -errorlogname=/tmp/object_restore.log
      ```
   b. When all restore jobs are completed, check the error logs. If any errors are found, correct them so that all restore operations are complete successfully.
8. Issue the following command to restore the object storage configuration data: **mmcesdr primary restore --file-config --restore**.
9. Verify that basic Swift commands (**swift stat** and **swift list**) return without error. Also, verify that the number of containers and the number of objects within those containers are as expected.

## Improving recovery time

The **dsmc restore** command starts a single restore job on a single node. This job might require a long period to restore all of your object data. To improve the restore performance, you can start separate restore jobs on different IBM Spectrum Protect client nodes.

You can create separate restore jobs by splitting a single restore task into several smaller ones. One way to do this is to specify the restore path for the object data that is deeper in the IBM Spectrum Scale object path.

For example, instead of starting the restore with the root of the IBM Spectrum Scale object path, start the object restore at the virtual devices level. If you have 40 virtual devices that are configured, you might start 40 independent restore jobs to restore the object data, and distribute the jobs to the different IBM Spectrum Protect client nodes. Additionally, you start a single restore job for all of the files under the account and container path.

With this approach, care must be taken not to overload the IBM Spectrum Protect client nodes or the IBM Spectrum Protect server. You might want to experiment to determine the best mix of jobs.

For example, if there are four IBM Spectrum Scale object nodes, each with the IBM Spectrum Protect client installed and configured, you might use the following types of commands:

1. On the first IBM Spectrum Scale object node, run a restore job for each of the first 10 virtual devices by running the following commands:

```
dsmc restore /gpfs0/objectfs/o/z1device0 -subdir=yes -disablenqr=no \
-servername=tsm1
dsmc restore /gpfs0/objectfs/o/z1device1 -subdir=yes -disablenqr=no \
-servername=tsm1
#<repeat for z1device2 - z1device9>
```

2. On the second node, run a restore job for each of the next 10 virtual devices. Continue the pattern on the remaining IBM Spectrum Scale object nodes so that all the virtual devices under the `o` subdirectory are restored. Also, start a single restore job for all the account and container data under the `ac` sub-directory by running the following command:

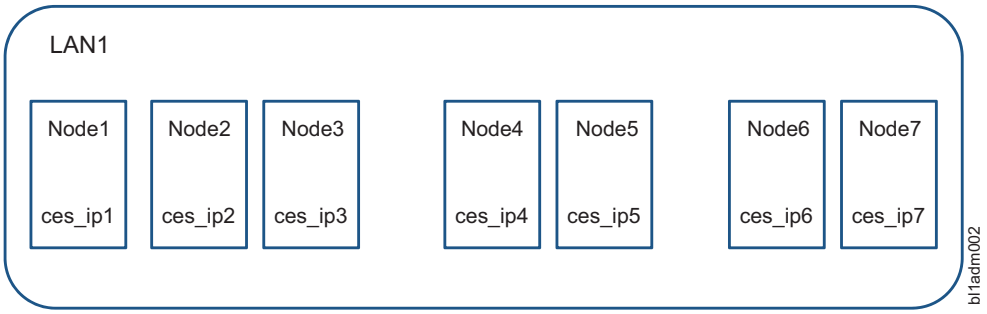
```
dsmc restore /gpfs0/objectfs/ac -subdir=yes -disablenqr=no -servername=tsm1
```

The most efficient restore approach depends on many factors, including the number of tape drives, IBM Spectrum Protect client configuration, and network bandwidth. You might need to experiment with your configuration to determine the most optimal restore strategy.

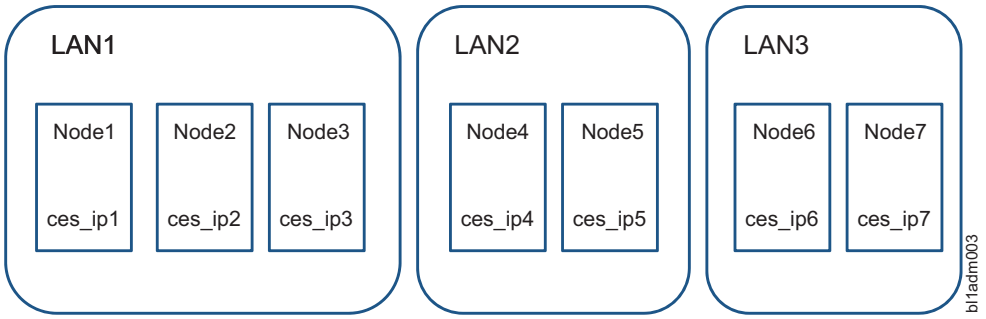
## Configuration of object for isolated node and network groups

You can configure object for isolated node and network groups.

Object needs constant network access between all the configured CES IP addresses. The standard configuration uses all the available CES IP addresses.




If a cluster configuration has isolated node and network groups and CES IP addresses have been assigned to those groups, parts of the object store are not accessible.



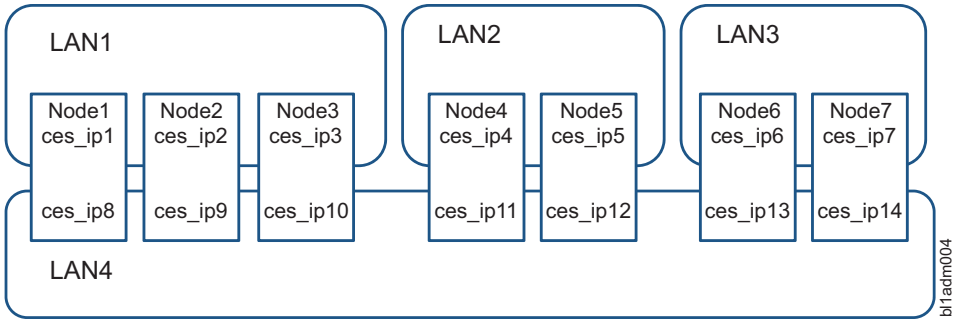

In such a configuration, a network and node group that must be used for object can be configured in the spectrum-scale-object.conf file. Only the CES IP addresses of this group are used for object.

**Note:** Only one object group can be used.

**Note:** If the Singleton and Database attributes of the IP assignments are changed manually by using the **mmces address change** command, only the IP addresses that belong to the object group can be used.

## Configuration example

In the following example, LAN1 is used as the object group, IP1, IP2, and IP3 are used for object, and the object store is fully available. If only LAN1 is used, the used object services will be limited to Node1, Node2, and Node3. To distribute the object load to all the nodes in the cluster, create a network that spans across all nodes. Create an object group and assign all nodes to it. Add new CES IPs, at least one CES IP per protocol node, and assign the IPs to the same Object Group.




1. To set up the object group and create the LAN4 group by adding nodes to groups, run the following command:

   ```
   mmchnode --ces-group=LAN4 -N Node1,Node2,Node3,Node4,Node5,Node6,Node7
   ```

   If you want to add CES IP addresses to the group, run the following command:

   ```
   mmces address add --ces-ip ces_ip8,ces_ip9,ces_ip10,ces_ip11,
   ces_ip12,ces_ip13,ces_ip14 --ces-group LAN4
   ```

   If you want to move the existing CES IP addresses to the group, run the following command:

   ```
   mmces address change --ces-ip ces_ip8,ces_ip9,ces_ip10,ces_ip11,
   ces_ip12,ces_ip13,ces_ip14 --ces-group LAN4
   ```

2. To set up the object group when object has already been configured, run the following command. The following command is on one line:

   ```
   mmobj config change --ccrfile spectrum-scale-object.conf --section node-group
   --property object-node-group --value LAN4
   ```

   To synchronize the ring files, run the following command:

   ```
   /usr/lpp/mmfs/bin/mmcesobjcrring --sync
   ```

3. To set up the object group when object has not been configured, use the `--ces-group` option of the **mmobj swift base** command:

   ```
   mmobj swift base  -g /gpfs/ObjectFS --cluster-hostname cluster-ces-ip.ibm --local-keystone
   --enable-s3 --admin-password Passw0rd --ces-group LAN4
   ```

   CES IPs ranging from ces_ip8 to ces_ip14 are used by the object store and the object load is distributed to all the nodes in the cluster. These IPs can be used for client connections, but more importantly, are used for traffic between the Swift proxy service and the account, container, and object services.

# Chapter 17. Managing GPFS quotas

The GPFS quota system helps you to control the allocation of files and data blocks in a file system.

GPFS quotas can be defined for:
- Individual users
- Groups of users
- Individual filesets

Quotas are enabled by the system administrator when control over the amount of space used by the individual users, groups of users, or individual filesets is required. By default, user and group quota limits are enforced across the entire file system. Optionally, the scope of quota enforcement can be limited to an individual fileset boundaries.

**GUI navigation**

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

Quota related tasks include:
1. "Enabling and disabling GPFS quota management"
2. "Default quotas" on page 228
3. "Explicitly establishing and changing quotas" on page 231
4. "Checking quotas" on page 234
5. "Listing quotas" on page 235
6. "Activating quota limit checking" on page 236
7. "Deactivating quota limit checking" on page 237
8. "Changing the scope of quota limit checking" on page 237
9. "Creating file system quota reports" on page 237
10. "Restoring quota files" on page 238

For GPFS fileset quotas, see "Filesets" on page 336.

**Note:** Windows nodes may be included in clusters that use GPFS quotas; however, Windows nodes do not support the quota commands.

## Enabling and disabling GPFS quota management

You can enable GPFS quota management on new or existing GPFS file systems, establish quota values, and disable quota management by following the steps in this topic.

To enable GPFS quota management on a new GPFS file system:
1. Specify the **-Q yes** option on the **mmcrfs** command. This option automatically activates quota enforcement whenever the file system is mounted. If you want the scope of quota limit enforcement to be based on individual filesets (rather than the entire file system), also specify the **--perfileset-quota** option on the **mmcrfs** command.
2. Mount the file system.
3. Issue the **mmedquota** or **mmsetquota** command to explicitly set quota values for users, groups, or filesets. See "Explicitly establishing and changing quotas" on page 231.

To enable GPFS quota management on an existing GPFS file system:

1. Run the **mmchfs -Q yes** command. This command automatically activates quota enforcement whenever the file system is mounted or activates all subsequent mounts following the new quota setting if the file system is not mounted. If you want the scope of quota limit enforcement to be based on individual filesets (rather than the entire file system), also specify the **--perfileset-quota** option on the **mmcrfs** command.

   If an online **mmchfs -Q yes/no** command fails or is interrupted for any reason, **mmcheckquota** or **mmchfs -Q yes/no** must be rerun so that quota configuration for all nodes in the cluster will be brought into a consistent state.

   All subsequent mounts will follow the new quota setting.

   **Note:** The **perfileset-quota** cannot be enabled online in GPFS 4.1.
2. Compile inode and disk block statistics using the **mmcheckquota** command. See "Checking quotas" on page 234. The values obtained can be used to establish realistic quota values when issuing the **mmedquota** or **mmsetquota** command.
3. Issue the **mmedquota** or **mmsetquota** command to explicitly set quota values for users, groups, or filesets. See "Explicitly establishing and changing quotas" on page 231.

Once GPFS quota management has been enabled, you may establish quota values by:

- Setting default quotas for all new users, groups of users, or filesets.
- Explicitly establishing or changing quotas for users, groups of users, or filesets.
- Using the **gpfs_quotactl()** subroutine.

To disable quota management, run the **mmchfs -Q no** command. All subsequent mounts will obey the new quota setting.

For complete usage information, see the *mmcheckquota command*, the *mmchfs command*, the *mmcrfs command*, and the *mmedquota command* in the *IBM Spectrum Scale: Command and Programming Reference*. For additional information on quotas, see the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Default quotas

Default quota limits can be set for new users, groups, and filesets for a specified file system. Default quota limits can also be applied at a more granular level for new users and groups in a specified fileset.

When default quotas are managed at the fileset level, those quotas have a higher priority than those set at the file system level. If the status of the fileset-level defaults for one fileset is **Initial**, they will inherit default limits from global fileset-level defaults. The status of newly added fileset-level default quotas can be one of the following:

**Initial** When the fileset is created, it will be in this state. All user and group quota accounts under the fileset will not follow the fileset defaults.

**Quota on**
All user and group quota accounts under the fileset that are created later will follow the fileset quota limits.

**Quota off**
All user and group quota accounts under the fileset that are created later will not follow the fileset quota limits. The users and groups will follow global fileset-level defaults if they are valid. If those defaults are not valid, the status will be initial.

Specific default quota recommendations for protocols:

- Since the protocols may have vastly different fileset requirements, it is not recommended to use default quotas at the fileset level. Rather, set explicit quotas and limits for each fileset in use by any and all protocols on a case-by-case basis.
- NFS: Prepare a default quota stanza file template, and at NFS export creation time, apply the default user or group quotas to the export path (assuming the export is an independent fileset) using per-fileset default quotas.
- SMB: Prepare a default quota stanza file template, and at SMB share creation time, apply the default user or group quotas to the share path (assuming the export is an independent fileset) using per-fileset default quotas.
- Object: IBM recommends using a single independent fileset, objectfs, for the object container. See *IBM Redpaper: A Deployment Guide for IBM Spectrum Scale Object* for details. With regard to quotas, here are the relevant sections from the Redpaper™:
  - GPFS quotas: The amount of disk space and the number of inodes that are assigned as upper limits for a specified user, group of users, or fileset. With OpenStack Swift, GPFS user quotas are not used; instead, the system relies on OpenStack Swift quotas to provide a similar type of service. However, GPFS fileset quotas can still be defined (for example, for inodes, to limit the resources that are consumed by the fileset). See *Chapter 1.3 Key concepts and terminology* of the IBM Redpaper for details.
  - Swift quotas: Allows specification of the amount of disk space or number of objects that can be consumed by either an account (and subsequently all of its containers) or to an individual container. The interaction between Swift quotas and GPFS quotas are described in more detail in *Chapter 6 Swift feature overview* and *Chapter 1.3 Key concepts and terminology* of the IBM Redpaper.
  - Quotas: Swift quotas allow a specific amount of disk capacity to be allocated to either containers or accounts by using Swift quotas. They also allow a limit on the maximum number of objects to be specified for containers or accounts. See *Chapter 6 Swift feature overview* of the IBM Redpaper for details.

    **Note:** Although GPFS quotas do not explicitly interact with Swift quotas, it still might be useful to employ GPFS quotas to limit the amount of space or the number of inodes that is consumed by the object store. To do this, define GPFS quotas on the top-level independent fileset by specifying the maximum size or maximum inode usage that the object store can consume. See *Chapter 6 Swift feature overview* of the IBM Redpaper for details.

To enable default quota values:

1. Ensure the file system is configured correctly to use quotas:
   a. The **-Q yes** option must be in effect for the file system.
   b. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect.

      **Note:** If **--perfileset-quota** is in effect, all users and groups in the fileset **root** will not be impacted by default quota unless they are explicitly set.

   The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options.
2. Enable default quotas with the **mmdefquotaon** command.
3. Specify default quota values for new users, groups, and filesets by issuing the **mmdefedquota** command using a default quota stanza file. A single invocation of the **mmsetquota** command using a quota stanza file can perform the following operations:
   - Set default user quotas on a file system.
   - Set default group quotas on a file system.
   - Set a default perfileset user quota on a fileset (if **--perfileset-quota** is in effect).

   The stanza file /tmp/defaultQuotaExample may look like this:

```
%quota:
 device=fs1
 command=setDefaultQuota
 type=USR
 blockQuota=25G
 blockLimit=30G
 filesQuota=10K
 filesLimit=11K
%quota:
 device=fs1
 command=setDefaultQuota
 type=GRP
 blockQuota=75G
 blockLimit=90G
 filesQuota=30K
 filesLimit=33K
%quota:
 device=fs1
 command=setDefaultQuota
 type=USR
 fileset=fset0
 blockQuota=25G
 blockLimit=30G
 filesQuota=10K
 filesLimit=11K
```

Then issue the command:

```
# mmsetquota –F /tmp/defaultQuotaExample
```

4. To activate quota checking, use the **mmquotaon** command.
5. To list quotas, use the **mmlsquota** command:

The default quotas can be deactivated by issuing the **mmdefquotaoff** command.

For fileset recommendations, see "Filesets and quotas" on page 338.

For complete usage information, see the *mmchfs command*, *mmcrfs command*, *mmdefedquota command*, *mmdefquotaoff command*, *mmdefquotaon command*, *mmedquota command*, *mmlsfs command*, and *mmsetquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Implications of quotas for different protocols

Quotas can mean different things for different protocols. This section describes how quotas affect the SMB and NFS protocols.

Quotas are stored and enforced in the file system. See Chapter 17, "Managing GPFS quotas," on page 227 for details on how to enable and use quotas.

- SMB protocol and quotas

  For SMB clients, quotas can limit the used and free space reported to clients:

  – If the SMB option **gpfs:dfreequota** is set, the user quota for the current user and the group quota for the user's primary group are queried during the free space query:
    - If the block limit is reached, the free space is reported as 0 and the size of the share is reported with the currently used data.
    - If the soft block quota is exceeded for longer than the block grace time, the free space is reported as 0 and the size of the share is reported with the currently used data.
    - If no limit is exceeded, the free space is reported as the free space according to the lowest quota limit.
    - If no quota is in place, the size and free space as queried from the underlying file system are reported.

- In the case of per-fileset user and group quotas, the quotas are only queried from the root folder of the export. If a subdirectory inside the share is in a different fileset, the user and group quotas are not considered for the free space report.

  **Note:** For including fileset quotas in the reported free space, configure the underlying file system with the --filesetdf flag (in **mmcrfs** or **mmchfs**). It is not possible to query or change individual quotas from a SMB client system.

- NFS protocol and quotas

  It is not possible to query or change individual quotas from a NFS client system. User and group quotas are not included in the reported free space to a client. To include fileset quotas in the reported space to a client, configure the underlying file system with the **--filesetdf** flag (in **mmcrfs** or **mmchfs**).

- Object protocol and quotas:
  - There are two applicable levels of quotas: The quotas in the file system (see Chapter 17, "Managing GPFS quotas," on page 227) and the quotas managed by Swift.
  - Swift quotas can be used as account and container quotas. The quota values are set as account and container metadata entries. For more information, see the OpenStack Swift documentation.
  - When using file system quotas, it is important to consider that all objects stored by Swift are stored with the same owner and owning group (swift:swift).

## Explicitly establishing and changing quotas

Use the **mmedquota** command to explicitly establish or change file system quota limits for users, groups of users, or filesets.

When setting quota limits for a file system, replication within the file system should be considered. See "Listing quotas" on page 235.

The **mmedquota** command opens a session using your default editor, and prompts you for soft and hard limits for blocks and inodes. For example, to set user quotas for user **jesmith**, enter:

```
mmedquota -u jesmith
```

The system displays information in your default editor similar to:

```
*** Edit quota limits for USR jesmith:
NOTE: block limits will be rounded up to the next multiple block size.
      block units may be: K, M, G, T or P, inode units may be: K, M or G.
gpfs0: blocks in use: 24576K, limits (soft = 0K, hard = 0K)
        inodes in use: 0, limits (soft = 0K, hard = 0K)
```

**Note:** A quota limit of zero indicates **no** quota limits have been established.

The current (in use) block and inode usage is for display only; it cannot be changed. When establishing a new quota, zeros appear as limits. Replace the zeros, or old values if you are changing existing limits, with values based on the user's needs and the resources available. When you close the editor, GPFS checks the values and applies them. If an invalid value is specified, GPFS generates an error message. If this occurs, reenter the **mmedquota** command. If the scope of quota limit enforcement is the entire file system, **mmedquota** will list all instances of the same user (for example, **jesmith**) on different GPFS file systems; if the quota enforcement is on a per-fileset basis, **mmedquota** will list all instances of the same user on different filesets on different GPFS file systems.

You may find it helpful to maintain a *quota prototype*, a set of limits that you can apply by name to any user, group, or fileset without entering the individual values manually. This makes it easy to set the same limits for all. The **mmedquota** command includes the **-p** option for naming a prototypical user, group, or fileset on which limits are to be based. The **-p** flag can only be used to propagate quotas from filesets within the same file system.

For example, to set group quotas for all users in a group named **blueteam** to the prototypical values established for **prototeam**, issue:

```
mmedquota -g -p prototeam blueteam
```

You may also reestablish default quotas for a specified user, group of users, or fileset when using the **-d** option on the **mmedquota** command.

**Note:** You can use the **mmsetquota** command as an alternative to the **mmedquota** command.

For complete usage information, see the *mmedquota command* and the *mmsetquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Setting quotas for users on a per-project basis

A file system must be properly configured in order to set quotas for users. Use this information to set quotas for any number of users on a per-project basis across protocols.

1. Ensure the file system is configured correctly to use quotas:
   a. The **-Q yes** option must be in effect for the file system.
   b. To set default quotas at the fileset level, the **--perfileset-quota** option must also be in effect.

      **Note:** If **--perfileset-quota** is in effect, all users and groups in the fileset **root** will not be impacted by default quota unless they are explicitly set.

   The **-Q yes** and **--perfileset-quota** options are specified when creating a file system with the **mmcrfs** command or changing file system attributes with the **mmchfs** command. Use the **mmlsfs** command to display the current settings of these quota options.

   Here are some examples:

   a. If a GPFS cluster is created with configuration profile file, `example.profile`, which contains the following lines:

      ```
      %filesystem
        quotasAccountingEnabled=yes
        quotasEnforced=user;group;fileset
        perfilesetQuotas=yes
      ```

      then, when a file system is created, those quota attributes will be set automatically: quota accounting will be enabled on a perfileset basis for users and groups, and quotas will automatically be enforced. This means that when a quota is reached, the end user will not be able to add more data to the file system.

      **mmcrfs fs5 nsd8**

      A listing of the file system config, using the **mmlsfs** command, will show the following attributes and values, having been set by the **mmcrfs** command:

      **mmlsfs fs5**

      ```
      ...
       -Q                    user;group;fileset       Quotas accounting enabled
                             user;group;fileset       Quotas enforced
                             none                     Default quotas enabled
       --perfileset-quota    Yes                      Per-fileset quota enforcement
       ....
      ```

      For more information on **mmcrcluster** user-defined profiles, see *mmcrcluster command* in the *IBM Spectrum Scale: Command and Programming Reference*.

   b. Whether or not a GPFS was created with a configuration profile file, a GPFS file system can be created with the quota attributes to be set. This can be done by calling the configuration profile file explicitly from the command line:

      **mmcrfs fs6 nsd9 --profile=example**

For more information on user-defined profiles, see *mmcrfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

2. Create a fileset on the file system for the project using the **mmcrfileset** command.

   For example:

   **mmcrfileset fs5 projectX --inode-space=new**

   **Note:** It is recommended to create an independent fileset for the project.

3. Link the fileset using the **mmlinkfileset** command.

   The file system, fs5, must be mounted, using the **mmmount** command. For example:

   **mmmount fs5 -a**

   **mmchfs fs5 --inode-limit 400000:300000**

   Output:

   ```
   Set maxInodes for inode space 0 to 400000
   Fileset root changed.
   ```

   **mmlinkfileset fs5 projectX -J /gpfs/fs5/projectX**

4. Create export/share using the newly created fileset as the export/share path. For more information, see the *mmnfs command* and the *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*. For example:

   **mmnfs export add /gpfs/fs5/projectX**

5. If needed, specify absolute fileset inode limits using the **mmchfileset** command. A fileset inode limit is analogous to saying this is how many files and directories the project is likely to produce. This is not something that can be easily recommended. Nonetheless, here is an example of how that can be set and listed for the file system and fileset:

   **mmchfileset fs5 projectX --inode-limit 200000**

   Output:

   ```
   Set maxInodes for inode space 1 to 200000
   Fileset projectX changed.
   ```

   **mmlsfileset fs5 -L**

   Output:

   ```
   Filesets in file
   system 'fs5':
      Name        Id   RootInode  ParentId   Created                   Inode Space   MaxInodes   AllocInodes  Comment
      root        0       3          --      Sat Mar 28 13:40:33 2015   0   400000    310656     root         fileset
      projectX    1     524291       --      Sat Mar 28 14:54:13 2015   1   200000    100032
   ```

6. Now that there is a fileset limit in place, which is entirely optional, to set group quota limits on the project, that is, on fileset **projectX** on file system **fs5**, use the **mmsetquota** command. For example, if the group **groupY** will access **projectX**:

   **mmsetquota fs5:projectX --group groupY --block 128G --files 150K**

   Here, the perfileset quota needs to be enabled on **fs5** as in step 1, and the group **groupY** must have a GID (group ID) on the GPFS cluster. The **block** parameter is used to specify the maximum size of the data on the storage device and the **files** parameter is used to specify the maximum number for files (or directories) the **groupY** is able to consume or create on **projectX**, a fileset of file system **fs5** that is exported through NFS in this example.

   At this point, the quota accounting needs to be refreshed on the file system using the **mmcheckquota** command, and then a reporting of the quota limits on **projectX** can take place using the **mmrepquota** command. For example:

   **mmcheckquota fs5**

   **mmrepquota fs5:projectX**

   Output:

```
                                Block Limits                            |          File Limits
Name    fileset    type  KB       quota  limit  in_doubt grace  |  files   quota  limit in_doubt  grace
root    projectX   USR    0           0      0         0   none  |      1       0      0         0   none
root    projectX   GRP    0           0      0         0   none  |      1       0      0         0   none
groupY  projectX   GRP    0   134217728      0         0   none  |      0  153600      0         0   none
```

7. If the project grows, or shrinks, and quota changes at the group level are needed, the **mmsetquota** command can again be used to change the quotas for **groupY** on **projectX**. For example, if the expected limits for **projectX doubles**:

   **mmsetquota fs5:projectX --group groupY --block 256G --files 300K**

   **mmrepquota fs5:projectX**

   Output:

```
                                Block Limits                            |          File Limits
Name   fileset    type   KB      quota  limit in_doubt  grace  |  files   quota  limit in_doubt  grace
root   projectX   USR     0          0      0         0   none  |      1       0      0         0   none
root   projectX   GRP     0          0      0         0   none  |      1       0      0         0   none
groupY projectX   GRP     0  268435456      0         0   none  |      0  307200      0         0   none
```

8. If the project is projected to exceed the inode limits for the fileset and file system, then these can also be adjusted upwards. For more information, see the *mmchfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Checking quotas

The **mmcheckquota** command counts inode and space usage for a file system and writes the collected data into quota files.

You must use the **mmcheckquota** command if any of the following are true:

1. Quota information is lost due to node failure.

   Node failure could leave users unable to open files or deny them disk space that their quotas should allow.

2. The *in doubt* value approaches the quota limit. To see the *in doubt* value, use the **mmlsquota** or **mmrepquota** commands.

   As the sum of the *in doubt* value and the current usage may not exceed the hard limit, the actual block space and number of files available to the user, group, or fileset may be constrained by the *in doubt* value. Should the *in doubt* value approach a significant percentage of the quota, use the **mmcheckquota** command to account for the lost space and files.

**Note:** Running **mmcheckquota** is also recommended (in an appropriate time slot) if the following message is output by **mmrepquota**, **mmlsquota**, or **mmedquota**:

```
Quota accounting information is inaccurate and quotacheck must be run.
```

When issuing the **mmcheckquota** command on a mounted file system, negative *in doubt* values may be reported if the quota server processes a combination of up-to-date and back-level information. This is a transient situation and may be ignored.

During the normal operation of file systems with quotas enabled (not running **mmcheckquota** online), the usage data reflects the actual usage of the blocks and inodes in the sense that if you delete files you should see the usage amount decrease. The *in doubt* value does not reflect how much the user has used already, it is just the amount of quotas that the quota server has assigned to its clients. The quota server does not know whether the assigned amount has been used or not. The only situation where the *in doubt* value is important to the user is when the sum of the usage and the *in doubt* value is greater than the user's quota hard limit. In this case, the user is not allowed to allocate more blocks or inodes unless he brings the usage down.

For example, to check quotas for the file system **fs1** and report differences between calculated and recorded disk quotas, enter:

```
mmcheckquota -v fs1
```

The information displayed shows that the quota information for **USR7** was corrected. Due to a system failure, this information was lost at the server, which recorded 0 subblocks and 0 files. The current usage data counted is 96 subblocks and 3 files. This is used to update the quota:

```
fs1: quota check found the following differences:
USR7: 96 subblocks counted (was 0); 3 inodes counted (was 0)
```

**Note:** In cases where small files do not have an additional block allocated for them, quota usage may show less space usage than expected.

For complete usage information, see the *mmcheckquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Listing quotas

The **mmlsquota** command displays the file system quota limits, default quota limits, and current usage information.

If the scope of quota limit enforcement is the entire file system, **mmlsquota -u** or **mmlsquota -g** will list all instances of the same user or group on different GPFS file systems. If the quota enforcement is on a per-fileset basis, **mmlsquota -u** or **mmlsquota -g** will list all instances of the same user or group on different filesets on different GPFS file systems.

GPFS quota management takes replication into account when reporting on and determining whether quota limits have been exceeded for both block and file usage. If either data or metadata replication is enabled, the values reported by both the **mmlsquota** command and the **mmrepquota** command may exceed the corresponding values reported by commands like **ls**, **du**, and so on. The difference depends on the level of replication and on the number of replicated file system objects. For example, if data block replication is set to 2, and if all files are replicated, then the reported block usage by the **mmlsquota** and **mmrepquota** commands will be double the usage reported by the **ls** command.

When the **mmlsquota** command is issued, negative *in doubt* values may be reported if the quota server processes a combination of up-to-date and back-level information. This is a transient situation and may be ignored.

Display the quota information for one user, group of users, or fileset with the **mmlsquota** command. If none of the options **-g**, **-u**, or **-j** are specified, the default is to display only user quotas for the user who issues the command.

To display default quota information, use the **-d** option with the **mmlsquota** command. For example, to display default quota information for users of all the file systems in the cluster, issue this command:

```
mmlsquota -d -u
```

The system displays information similar to:

```
         Default Block Limits(KB)          | Default File Limits
Filesystem type          quota      limit  |    quota    limit Remarks
fs1        USR         5242880    6291456  |        0        0

         Default Block Limits(KB)          | Default File Limits
Filesystem type          quota      limit  |    quota    limit Remarks
fs2        USR         no default limits
```

In this example, file system **fs1** shows that the default block quota for users is set at 5 GB for the soft limit and 6 GB for the hard limit. For file system **fs2**, no default quotas for users have been established.

When **mmlsquota -d** is specified in combination with the **-u**, **-g**, or **-j** options, default file system quotas are displayed. When **mmlsquota -d** is specified without any of the **-u**, **-g**, or **-j** options, default fileset-level quotas are displayed.

If you issue the **mmlsquota** command with the **-e** option, the quota system collects updated information from all nodes before returning output. If the node to which *in-doubt* space was allocated should fail before updating the quota system about its actual usage, this space might be lost. Should the amount of space in doubt approach a significant percentage of the quota, run the **mmcheckquota** command to account for the lost space.

To collect and display updated quota information about a group named **blueteam**, specify the **-g** and **-e** options:

```
mmlsquota -g blueteam -e
```

The system displays information similar to:

```
                  Block Limits                                | File Limits
Filesystem type     KB       quota   limit  in_doubt  grace | files quota   limit   in_doubt  grace
Disk quotas for group blueteam (gid 100):
fs1        GRP   45730       52000   99000      1335   none | 411    580      990          19   none
```

For complete usage information, see the *mmlsquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Activating quota limit checking

Quota limit checking can be activated for users, groups, filesets, or any combination of these three.

You can have quotas activated automatically whenever the file system is mounted by specifying the quota option (**-Q yes**) when creating (**mmcrfs -Q yes**) or changing (**mmchfs -Q yes**) a GPFS file system. When creating a file system, the default is to **not** have quotas activated, so you must specify this option if you want quotas activated.

The **mmquotaon** command is used to turn quota limit checking back on if it had been deactivated by issuing the **mmquotaoff** command. Specify the file system name, and whether user, group, or fileset quotas are to be activated. If you want all three fileset quotas activated (user, group, and fileset), specify only the file system name. After quotas have been turned back on, issue the **mmcheckquota** command to count inode and space usage.

For example, to activate user quotas on the file system **fs1**, enter:

```
mmquotaon -u fs1
```

To confirm the change, enter:

```
mmlsfs fs1 -Q
```

The system displays output similar to:

```
flag value          description
---- -------------- -----------------------------------------------------
 -Q  user           Quotas enforced
```

For complete usage information, see the *mmquotaon command* and the *mmlsfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deactivating quota limit checking

During normal operation, there is no need to deactivate quota enforcement. The only reason you might have to deactivate quota enforcement is when users are denied allocation that their quotas should allow, due to loss of quota information during node failure.

If this occurs, use the **mmcheckquota** command after reactivating quotas to reconcile allocation data. When quota enforcement is deactivated, disk space and file allocations are made without regard to limits.

The **mmquotaoff** command is used to deactivate quota limit checking. Specify the file system name and whether user, group, or fileset quotas, or any combination of these three, are to be deactivated. If you want all types of quotas deactivated, specify only the file system name.

For example, to deactivate only user quotas on the file system **fs1**, enter:

```
mmquotaoff -u fs1
```

To confirm the change, enter:

```
mmlsfs fs1 -Q
```

The system displays output similar to:

```
flag value          description
---- -------------- --------------------
 -Q  group;fileset  Quotas enforced
```

For complete usage information, see the *mmquotaoff command* and the *mmlsfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Changing the scope of quota limit checking

The scope of quota enforcement is established when quotas are activated. By default, user and group quota limits are enforced across the entire file system. Optionally, the scope of quota enforcement can be limited to an individual fileset boundaries.

The scope of quota enforcement can be changed using the **mmchfs** command and specifying either the **--perfileset-quota** or **--noperfileset-quota** option as needed.

After changing the scope of quota enforcement, **mmcheckquota** must be run to properly update the quota usage information.

## Creating file system quota reports

You can have GPFS prepare a quota report for a file system using the **mmrepquota** command.

The quota report lists:

1. Number of files used
2. Amount of disk space used
3. Current quota limits
4. In doubt quotas (disk space allocated but currently unaccounted for)
5. Grace period allowance to exceed the soft limit
6. Whether the quotas have been explicitly set (**e**), are default values at the file system level (**d_fsys**), are default values at the fileset level (**d_fset**), or initial values (**i**)

   The entry type also indicates whether or not default quotas are enabled for the file system (**default on** or **default off**).

Specify whether you want to list only user quota information (**-u** flag), group quota information (**-g** flag), or fileset quota information (**-j** flag) in the **mmrepquota** command. The default is to summarize all three quotas. If the **-e** flag is not specified, there is the potential to display negative usage values as the quota server may process a combination of up-to-date and back-level information. See "Listing quotas" on page 235.

If the scope of quota limit enforcement is the entire file system, **mmrepquota -u** or **mmrepquota -g** will list all users or groups on different GPFS file systems. If the quota enforcement is on a per-fileset basis, **mmrepquota -u** or **mmrepquota -g** will list all instances of the same user or group on different filesets on different GPFS file systems.

To list the group quotas (**-g** option) for all file systems in the cluster (**-a** option), and print a report with header lines (**-v** option), enter:

```
mmrepquota -g -v -a
```

The system displays information similar to:

```
*** Report for GRP  quotas on fs1
              Block Limits                                |                File Limits
Name    type     KB  quota  limit  in_doubt  grace      |  files  quota  limit in_doubt   grace entryType
system  GRP   25088      0      0    209120   none      |     32      0      0     1078    none default on
usr     GRP  435256      0      0    199712   none      |     11      0      0      899    none d_fsys
```

For complete usage information, see the *mmrepquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Restoring quota files

The method that is used for restoring GPFS quota files depends on the version of GPFS.

The three scenarios for restoring GPFS quota files follow.

1. The file system version is lower than `4.1.0.0`.

   In scenario 1, quota files can be backed up directly by copying visible quota files and then restored using the **mmcheckquota** command. The newly-specified backup quota files are transferred from normal files to quota files (metadata). Old quota files are converted from metadata to "normal" files, so these old quota files can be deleted.
2. The file system version is `4.1.0.0` or higher, but lower than `4.1.1.0`.

   In scenario 2, quota files cannot be restored using the **mmcheckquota** command.
3. The file system version is `4.1.1.0` (or higher).

   In scenario 3, quota files can be restored using the **mmcheckquota** command. Use the **mmcheckquota --backup** command to back up quota files. You can restore quota files from the former backup quota files. The **mmcheckquota** command copies data from specified backup quota files to "invisible" quota files. You cannot view or delete the original quota files. You can delete specified backup quota files only.

Additional details about the three scenarios for restoring GPFS quota files follow.

In scenarios 1 and 3:

- User, group, and fileset quota files can be restored from a backup copy of the original quota file. When restoring quota files, the backup file must be in the root directory of the GPFS file system.

  In scenario 1, if a backup copy of the original quota file does not exist, an empty file will be created when the **mmcheckquota** command is issued.

  In scenario 3, the **mmcheckquota** command does nothing and prints an error.
- The user, group, or fileset files can be restored from backup copies by issuing the **mmcheckquota** command with the appropriate options.

1. To restore the user quota file for the file system **fs1** from the backup file **userQuotaInfo**, enter:
   ```
   mmcheckquota -u userQuotaInfo fs1
   ```

   This command must be run offline (that is, no nodes are mounted).
2. This will restore the user quota limits set for the file system, but the usage information will not be current. To bring the usage information to current values, the command must be reissued:
   ```
   mmcheckquota fs1
   ```

In scenario 1, if you want to nullify all quota configuration and then reinitialize it, follow these steps:

1. Remove the existing quota files that are corrupted.
   a. Disable quota management:
      ```
      mmchfs fs1 -Q no
      ```
   b. Remove the **user.quota**, **group.quota**, and **fileset.quota** files.
2. Enable quota management.
   a. Issue the following command:
      ```
      mmchfs fs1 -Q yes
      ```
3. Reestablish quota limits by issuing the **mmedquota** command or the **mmdefedquota** command.
4. Gather the current quota usage values by issuing the **mmcheckquota** command.

In scenario 2, quota files do not exist externally. Therefore, use **mmbackupconfig** and **mmrestoreconfig** to restore quota configurations.

For complete usage information, see the *mmcheckquota command*, the *mmdefedquota command*, and the *mmedquota command* in the *IBM Spectrum Scale: Command and Programming Reference*.

# Chapter 18. Managing GUI administrators

GUI administrators of the IBM Spectrum Scale system can monitor, configure, and manage the IBM Spectrum Scale system and are distinguished from the data users.

You can manage GUI users either locally within the system or in an external authentication server such as Microsoft Active Directory (AD) or Lightweight Directory Access Protocol Server (LDAP). By default, the IBM Spectrum Scale system uses an internal authentication repository for GUI users. To use an external AD or LDAP server, you need to disable the internal user repository that is used for the GUI user management and enable the LDAP/AD repository. For more information on how to disable internal repository and enable the external repository, see "Managing GUI administrators in an external authentication server" on page 243.

## Managing administrative users locally in the IBM Spectrum Scale system

You can create users who can perform different administrative tasks on the system. Each user must be part of a user group or multiple groups that are defined on the system. When you create a new user, you assign the user to one of the default user groups or to a custom user group. User groups are assigned with predefined roles that authorize the users within that group to a specific set of operations on the GUI.

Use the **Access** > **GUI Users** page to create users and add them to a user group.

Predefined roles are assigned to user groups to define the working scope within the GUI. If a user is assigned to more than one user group, the permissions are additive, not restrictive. The predefined role names cannot be changed.

The following are the default user groups:

- **Administrator**

  Manages all functions on the system except those deals with managing users, user groups, and authentication.
- **SecurityAdmin**

  Manages all functions on the system, including managing users, user groups, and user authentication.
- **SystemAdmin**

  Manages clusters, nodes, alert logs, and authentication.
- **StorageAdmin**

  Manages disks, file systems, pools, filesets, and ILM policies.
- **SnapAdmin**

  Manages snapshots for file systems and filesets.
- **DataAccess**

  Controls access to data. For example, managing access control lists.
- **Monitor**

  Monitors objects and system configuration but cannot configure, modify, or manage the system or its resources.
- **ProtocolAdmin**

  Manages object storage and data export definitions of SMB and NFS protocols.
- **UserAdmin**

Manages access for GUI users. Users who are part of this group have edit permissions only in the Access pages of the GUI.

The IBM Spectrum Scale system is delivered with a default GUI user named *admin*. This user is also stored in the local repository. You can log in to the system by using this user name to create additional GUI users and groups in local user repository.

Use the various controls that are available under the **Password Policy** tab of the GUI Users page to enforce strong passwords for the users. You can modify or expire password of the individual users or all the users that are created in the system. If the password is set as expired, the users will be prompted to change the password in the next login.

Use the various controls that are available under the Password Policy tab of the GUI Users page to enforce strong passwords for the users. You can modify or expire password of the individual users or all the users that are created in the system. If the password is set as expired, the users will be prompted to change the password in the next login.

**User groups**

Users who are part of Security Administrator and User Administrator user groups can create role-based user groups where any users that are added to the group adopt the role that is assigned to that group.

Roles apply to users on the system and are based on the user group to which the user belongs. A user can be part of multiple user groups so that a single user can play multiple roles in the system. You can assign the following roles to your user groups:

- **Administrator**

  Users can access all functions on the GUI except those deals with managing users and user groups.
- **Security Administrator**

  Users can access all functions on the GUI, including managing users and user groups.
- **System Administrator**

  Users manage clusters, nodes, and alert logs.
- **Storage Administrator**

  Users manage disks, file systems, pools, and filesets.
- **Snapshot Administrator**

  Users manage snapshots for file systems, filesets.
- **Monitor**

  Users can view objects and system configuration but cannot configure, modify, or manage the system or its resources.
- **Data Access**

  Users can perform the following tasks:
  - Edit owner, group, and ACL of any file or path through the **Access** > **File System ACL** > **Files and Directories** page.
  - Edit owner, group, and ACL for a non-empty directory of a file system, fileset, NFS export, or SMB share.
  - Create or delete object containers through the **Object** > **Accounts** page.
- **Protocol Administrator**

  Users manage object storage and data export definitions of SMB and NFS protocols.
- **User Administrator**

  Users manage GUI users and user groups.

**Note:** Only users with *User Administrator* role can modify the password policy of a user.

**Note:** Default groups are not created for the user role *User Administrator* in case the user is upgrading the IBM Spectrum Scale cluster from 4.2.0.x to 4.2.1.

## Managing GUI administrators in an external authentication server

By default, the IBM Spectrum Scale uses an internal authentication repository for the GUI administrators. To use an external AD or LDAP server to manage the GUI administrators, perform the following steps:

1. Disable the internal user repository by performing the following steps:
   a. Access the server that is running the GUI and open the following file by using a text editor: `/opt/ibm/wlp/usr/servers/gpfsgui/server.xml`
   b. Comment out the two elements that are referring to the `FsccUserRepo/Registry` that are highlighted in boldface in the following example:

```
<server description="GSS GUI">
<featureManager>
<feature>jsp-2.2</feature>
<feature>localConnector-1.0</feature>
<feature>jdbc-4.0</feature>
<feature>ssl-1.0</feature>
<feature>servlet-3.0</feature>
<feature>appSecurity-2.0</feature>
<!-- <feature>usr:FsccUserRepo</feature> -->
<feature>jndi-1.0</feature>
</featureManager>
<!-- <fsccUserRegistry prefFile="${server.config.dir}/preferences.xml"/> -->
[...]
```

2. Add the LDAP or AD feature to the WebSphere® Liberty to enable LDAP or AD support in WebSphere Liberty by performing the following steps:
   a. Access the `server.xml` file, which is at the following location: `/opt/ibm/wlp/usr/servers/gpfsgui/server.xml`
   b. Add the entry `ldapRegistry-3.0` in the `server.xml` as shown in the following example:

```
<server description="GSS GUI">
<featureManager>
<feature>jsp-2.2</feature>
<feature>localConnector-1.0</feature>
<feature>jdbc-4.0</feature>
<feature>ssl-1.0</feature>
<feature>servlet-3.0</feature>
<feature>appSecurity-2.0</feature>
<!-- <feature>usr:FsccUserRepo</feature> -->
<feature>jndi-1.0</feature>
<feature>ldapRegistry-3.0</feature>
</featureManager>
```

   **Note:** When the IBM Spectrum Scale system is updated to the latest release, the *server.xml* is overwritten as part of the update. Therefore, if an external authentication server is used for managing GUI administrators, the *server.xml* file must be edited accordingly after every system update.

3. Configure *<ldapRegistry>* element in the LDAP or AD repository. Depending on the type of the external server, the configuration element can have different attribute values. Sample configurations for AD and IBM Directory Server are given in the following example:

   **Active Directory Server**

```
<ldapRegistry id="ldap"
    host="ldapserver.mycity.mycompany.com" port="389" ignoreCase="true"
    baseDN="cn=users,dc=adtest,dc=mycity,dc=mycompany,dc=com"
    bindDN="cn=testuser,cn=users,dc=adtest,dc=mycity,dc=mycompany,dc=com"
    bindPassword="testuserpwd"
    ldapType="Microsoft Active Directory"
    sslEnabled="false">
<actived Filters
userFilter="(&(sAMAccountName=%v)(objectcategory=person))"
```

```
 groupFilter="(&(cn=%v)(objectcategory=group))"
userIdMap="user:sAMAccountName"
groupIdMap="*:cn"
groupMemberIdMap="memberOf:member">
</actived Filters>
</ldapRegistry>
```

**IBM Directory Server:**

```
<ldapRegistry id="ldap"
    host="ldapserver.mycity.mycompany.com" port="389" ignoreCase="true"
    baseDN="o=mycompany,c=us"
    ldapType="IBM Tivoli Directory Server"
    sslEnabled="false">
<idsFilters
   userFilter="(&(uid=%v)(objectclass=ePerson))"
groupFilter="(&(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfURLs)))"
userIdMap="*:uid"
groupIdMap="*:cn"
groupMemberIdMap="mycompany-allGroups:member;mycompany-allGroups:uniqueMember;
groupOfNames:member;groupOfUniqueNames:uniqueMember">
</idsFilters>
</ldapRegistry>
```

For more information on the advanced configuration options or for enabling SSL, see Configuring LDAP user registries in Liberty.

4. Establish the LDAP group to GUI role mapping. After the GUI server was restarted, you must review the groups to roles mapping and add/remove group to role mappings as necessary.
5. View and modify the existing group to role mappings. You can view the existing groups by using the **lsusergrp** command. Adding and removing groups can be done by using **mkusergrp** and **rmusergrp** respectively.

   **Note:** The commands that are used to manage the GUI administrators are not available in the same path where all other IBM Spectrum Scale commands are located. The GUI user management commands are located at the following location in the system: `/usr/lpp/mmfs/gui/cli`
6. Create a group to role mapping for initial access. For initial GUI access, you need to map one existing LDAP or AD group to the SecurityAdmin GUI role. The group name needs to match the CN attribute of the corresponding group in the external LDAP or AD repository. Log on to the server that is hosting the GUI and run the following command, which maps the specified LDAP group to the GUI role SecurityAdmin.

   ```
   # /usr/lpp/mmfs/gui/cli/mkusergrp mySecurityAdminLDAPGroup --role securityadmin
   ```
7. After the initial setup, any additional group mappings can be created through the GUI by using the **Create Group Mapping** option that is available in the **Access** > **GUI Access** page of the IBM Spectrum Scale management GUI.

   **Note:** The GUI Access page is available only if an external authentication server is enabled to manage the GUI user authentication. If an internal user repository is used for GUI user authentication, the GUI displays GUI Users page to create and manage GUI users and user roles.

# Chapter 19. Managing GPFS access control lists

Access control protects directories and files by providing a means of specifying who is granted access. GPFS access control lists are either traditional ACLs based on the POSIX model, or NFS V4 ACLs. NFS V4 ACLs are very different than traditional ACLs, and provide much more fine control of file and directory access. A GPFS file system can also be exported using NFS.

Management of GPFS access control lists (ACLs) and NFS export includes these topics:



## Traditional GPFS ACL administration

Support for NFS V4 access control lists (ACLs) has been added to traditional ACL support. NFS V4 ACLs are very different than the traditional ones.

If you are using NFS V4 ACLs, see "NFS V4 ACL administration" on page 249. Both ACL types may coexist in a single GPFS file system.

Traditional GPFS ACLs are based on the POSIX model. Traditional GPFS access control lists (ACLs) extend the base permissions, or standard file access modes, of read (r), write (w), and execute (x) beyond the three categories of file owner, file group, and other users, to allow the definition of additional users and user groups. In addition, GPFS introduces a fourth access mode, control (c), which can be used to govern who can manage the ACL itself.

In this way, a traditional ACL can be created that looks like this:

```
#owner:jesmith
#group:team_A
user::rwxc
group::rwx-
other::--x-
mask::rwxc
user:alpha:r-xc
group:audit:r-x-
group:system:rwx-
```

In this ACL:

- The first two lines are comments showing the file's owner, **jesmith**, and group name, **team_A**
- The next three lines contain the base permissions for the file. These three entries are the minimum necessary for a GPFS ACL:
  1. The permissions set for the file owner (**user**), **jesmith**
  2. The permissions set for the owner's **group**, **team_A**
  3. The permissions set for **other** groups or users outside the owner's group and not belonging to any named entry
- The next line, with an entry type of **mask**, contains the maximum permissions allowed for any entries other than the owner (the **user** entry) and those covered by **other** in the ACL.
- The last three lines contain additional entries for specific users and groups. These permissions are limited by those specified in the mask entry, but you may specify any number of additional entries up to a memory page (approximately 4 K) in size.

Traditional GPFS ACLs are fully compatible with the base operating system permission set. Any change to the base permissions, using the **chmod** command, for example, modifies the corresponding GPFS ACL as well. Similarly, any change to the GPFS ACL is reflected in the output of commands such as **ls -l**. Note that the control (c) permission is GPFS specific. There is no comparable support in the base operating system commands. As a result, the (c) permission is visible only with the GPFS ACL commands.

Each GPFS file or directory has an *access ACL* that determines its access privileges. These ACLs control who is allowed to read or write at the file or directory level, as well as who is allowed to change the ACL itself.

In addition to an *access ACL*, a directory may also have a *default ACL*. If present, the default ACL is used as a base for the access ACL of every object created in that directory. This allows a user to protect all files in a directory without explicitly setting an ACL for each one.

When a new object is created, and the parent directory has a default ACL, the entries of the default ACL are copied to the new object's access ACL. After that, the base permissions for user, mask (or group if mask is not defined), and other, are changed to their intersection with the corresponding permissions from the mode parameter in the function that creates the object.

If the new object is a directory, its default ACL is set to the default ACL of the parent directory. If the parent directory does not have a default ACL, the initial access ACL of newly created objects consists only of the three required entries (user, group, other). The values of these entries are based on the mode parameter in the function that creates the object and the umask currently in effect for the process.

Administrative tasks associated with traditional GPFS ACLs are:

1. "Setting traditional GPFS access control lists"
2. "Displaying traditional GPFS access control lists" on page 247
3. "Changing traditional GPFS access control lists" on page 248
4. "Deleting traditional GPFS access control lists" on page 248

## Setting traditional GPFS access control lists

Use the following information to set GPFS ACLs:

**GUI navigation**

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

Use the **mmputacl** command to set the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to set the ACL for a file named **project2.history**, we can create a file named **project2.acl** that contains:

```
user::rwxc
group::rwx-
other::--x-
mask::rwxc
user:alpha:r-xc
group:audit:rw--
group:system:rwx-
```

In this example,

- The first three lines are the required ACL entries setting permissions for the file's owner, the owner's group, and for processes that are not covered by any other ACL entry.
- The last three lines contain named entries for specific users and groups.

- Because the ACL contains named entries for specific users and groups, the fourth line contains the required mask entry, which is applied to all named entries (entries other than the **user** and **other**).

Once you are satisfied that the correct permissions are set in the ACL file, you can apply them to the target file with the **mmputacl** command. For example, to set permissions contained in the file **project2.acl** for the file **project2.history**, enter:

```
mmputacl -i project2.acl project2.history
```

To confirm the changes, enter:

```
mmgetacl project2.history
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwxc
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwxc #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

Although you can issue the **mmputacl** command without using the **-i** option to specify an ACL input file, and make ACL entries through standard input, you will probably find the **-i** option more useful for avoiding errors when creating a new ACL.

For complete usage information, see the *mmputacl command* and the *mmgetacl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Displaying traditional GPFS access control lists

Use the **mmgetacl** command to display the access ACL of a file or subdirectory, or the default ACL of a directory. For example, to display the ACL for the file **project2.history**, enter:

```
mmgetacl project2.history
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwxc
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwxc #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

The first two lines are comments displayed by the **mmgetacl** command, showing the owner and owning group. All entries containing permissions that are not allowed (because they are not set in the mask entry) display with a comment showing their effective permissions.

For complete usage information, see the *mmgetacl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Applying an existing traditional GPFS access control list

To apply the same traditional ACLs from one file or directory to another:

1. Issue the **mmgetacl** command with the **-o** option to place the information in an output file.
2. Apply the ACLs to the new file or directory by issuing the **mmputacl** command with the **-i** option.

For example, use the **-o** option to specify a file to which the ACL is written:

```
mmgetacl -o old.acl project2.history
```

Then, to assign the same permissions to another file, **project.notes**, enter:

```
mmputacl -i old.acl project.notes
```

To confirm the changes, enter:

```
mmgetacl project.notes
```

The information sent to standard output is similar to:

```
#owner:guest
#group:usr
user::rwxc
group::rwx- #effective:rw--
other::--x-
mask::rw-c
user:alpha:rwxc #effective:rw-c
group:audit:rwx- #effective:rw--
group:system:-w--
```

For complete usage information, see the *mmgetacl command* and the *mmputacl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Changing traditional GPFS access control lists

Use the **mmeditacl** command to change or create the traditional ACL of a file or directory, or the default ACL of a directory. For example, to interactively edit the ACL for the file **project2.history**, enter:

```
mmeditacl project2.history
```

The current ACL entries are displayed using the default editor, provided that the EDITOR environment variable specifies a complete path name. When the file is saved, the system displays information similar to:

```
mmeditacl: 6027-967 Should the modified ACL be applied? (yes) or (no)
```

After responding **yes**, the ACLs are applied.

For complete usage information, see the *mmeditacl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting traditional GPFS access control lists

Use the **mmdelacl** command to delete the extended entries in a traditional ACL of a file or directory, or the default ACL of a directory. For example, to delete the ACL for the directory **project2**, enter:

```
mmdelacl project2
```

To confirm the deletion, enter:

```
mmgetacl project2
```

The system displays information similar to:

```
#owner:uno
#group:system
user::rwxc
group::r-x-
other::--x-
```

You cannot delete the base permissions. These remain in effect after this command is executed.

For complete usage information, see the *mmdelacl command* and the *mmgetacl command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## NFS V4 ACL administration

AIX does not allow a file system to be NFS V4 exported unless it supports NFS V4 ACLs. By contrast, Linux does not allow a file system to be NFS V4 exported unless it supports POSIX ACLs.

This is because NFS V4 Linux servers handle NFS V4 ACLs by translating them into POSIX ACLs. For more information, see "Linux ACLs and extended attributes" on page 276.

**Note:**

This topic does not refer to the NFS Server function included with CES. For information, see "Authorizing protocol users" on page 258.

With AIX, the file system must be configured to support NFS V4 ACLs (with the **-k all** or **-k nfs4** option of the **mmcrfs** or **mmchfs** command). The default for the **mmcrfs** command is **-k all**.

With Linux, the file system must be configured to support POSIX ACLs (with the **-k all** or **-k posix** option of the **mmcrfs** or **mmchfs** command).

Depending on the value (**posix** | **nfs4** | **all**) of the **-k** parameter, one or both ACL types can be allowed for a given file system. Since ACLs are assigned on a per-file basis, this means that within the same file system one file may have an NFS V4 ACL, while another has a POSIX ACL. The type of ACL can be changed by using the **mmputacl** or **mmeditacl** command to assign a new ACL or by the **mmdelacl** command (causing the permissions to revert to the mode which is in effect a POSIX ACL). At any point in time, only a single ACL can be associated with a file. Access evaluation is done as required by the ACL type associated with the file.

NFS V4 ACLs are represented in a completely different format than traditional ACLs. For detailed information on NFS V4 and its ACLs, refer to *NFS Version 4 Protocol* and other information found in the Network File System Version 4 (nfsv4) section of the IETF Datatracker website (datatracker.ietf.org/wg/nfsv4/documents).

In the case of NFS V4 ACLs, there is no concept of a default ACL. Instead, there is a single ACL and the individual ACL entries can be flagged as being *inherited* (either by files, directories, both, or neither). Consequently, specifying the **-d** flag on the **mmputacl** command for an NFS V4 ACL is an error.

## NFS V4 ACL Syntax

An NFS V4 ACL consists of a list of ACL entries. Where traditional ACLs can display one entry per line, the GPFS representation of NFS V4 ACL entries are three lines each, due to the increased number of available permissions beyond the traditional **rwxc**.

The first line has several parts separated by colons (':').

- The first part identifies the user or group.
- The second part displays a **rwxc** translation of the permissions that appear on the subsequent two lines.
- The third part is the ACL type. NFS V4 provides both an *allow* and *deny* type.

  *allow* Means to allow (or permit) those permissions that have been selected with an 'X'.

  *deny* Means to not allow (or deny) those permissions that have been selected with an 'X'.
- The fourth and final part is a list of flags indicating *inheritance*.

  Valid flag values are:

**DirInherit**
Indicates that the ACL entry should be included in the initial ACL for subdirectories created in this directory (as well as the current directory).

**FileInherit**
Indicates that the ACL entry should be included in the initial ACL for files created in this directory.

**Inherited**
Indicates that the current ACL entry was derived from inherit entries in an NFS v4 ACL of the parent directory.

**InheritOnly**
Indicates that the current ACL entry should *not* apply to the directory, but *should* be included in the initial ACL for objects created in this directory.

**NoPropagateInherit**
Indicates that the ACL entry should be included in the initial ACL for subdirectories created in this directory but not further propagated to subdirectories created below *that* level.

As in traditional ACLs, users and groups are identified by specifying the type and name. For example, **group:staff** or **user:bin**. NFS V4 provides for a set of special names that are not associated with a specific local UID or GID. These special names are identified with the keyword **special** followed by the NFS V4 name. These names are recognized by the fact that they end with the character '@'. For example, **special:owner@** refers to the owner of the file, **special:group@** the owning group, and **special:everyone@** applies to all users.

The next two lines provide a list of the available access permissions that may be *allowed* or *denied*, based on the ACL type specified on the first line. A permission is selected using an 'X'. Permissions that are not specified by the entry should be left marked with '-' (minus sign).

These are examples of NFS V4 ACLs.

1. An ACL entry that explicitly allows **READ**, **EXECUTE** and **READ_ATTR** to the **staff** group on a file is similar to this:

```
group:staff:r-x-:allow
 (X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL  (X)READ_ATTR  (-)READ_NAMED
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

2. A Directory ACL is similar to this. It may include *inherit* ACL entries that do not apply to the directory itself, but instead become the initial ACL for any objects created within the directory.

```
special:group@:----:deny:DirInherit:InheritOnly
 (X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL  (X)READ_ATTR  (-)READ_NAMED
 (-)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

3. A complete NFS V4 ACL is similar to this:

```
#NFSv4 ACL
#owner:smithj
#group:staff
special:owner@:rwxc:allow:FileInherit
 (X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR  (-)READ_NAMED
 (X)DELETE    (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (-)WRITE_NAMED

special:owner@:rwxc:allow:DirInherit:InheritOnly
 (X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR  (-)READ_NAMED
 (X)DELETE    (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

user:smithj:rwxc:allow
 (X)READ/LIST (X)WRITE/CREATE (X)APPEND/MKDIR (-)SYNCHRONIZE (X)READ_ACL  (X)READ_ATTR  (-)READ_NAMED
 (X)DELETE    (X)DELETE_CHILD (X)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED
```

### ACL entries DELETE and DELETE_CHILD

The ACL entries **DELETE** and **DELETE_CHILD** require special considerations. The effect of various combinations of the **DELETE** attribute for a file, and the **DELETE_CHILD** attribute for its parent directory, is given in Table 27.

In this table, the columns refer to the ACL entry for a given file, and the rows refer to the ACL entry for its parent directory. The various combinations of these attributes produce one of these results:

**Permit**
: Indicates that GPFS permits removal of a file with the combination of file and parent directory ACL entries specified. (Other permission checking may exist within the operating system as well.)

**Deny**
: Indicates that GPFS denies (does not permit) removal of a file with the combination of file and parent directory ACL entries specified.

Removal of a file includes renaming the file, moving the file from one directory to another even if the file name remains the same, and deleting it.

*Table 27. Removal of a file with ACL entries DELETE and DELETE_CHILD*

| | **ACL Allows DELETE** | **ACL Denies DELETE** | **DELETE not specified** | **UNIX mode bits only** |
|---|---|---|---|---|
| **ACL Allows DELETE_CHILD** | Permit | Permit | Permit | Permit |
| **ACL Denies DELETE_CHILD** | Permit | Deny | Deny | Deny |
| **DELETE_CHILD not specified** | Permit | Deny | Deny | Deny |
| **UNIX mode bits only - wx permissions allowed** | Permit | Permit | Permit | Permit |
| **UNIX mode bits only - no w or no x permissions allowed** | Permit | Deny | Deny | Deny |

The UNIX mode bits are used in cases where the ACL is not an NFS V4 ACL.

## NFS V4 ACL translation

NFS V4 access requires that an NFS V4 ACL be returned to clients whenever the ACL is read. This means that if a traditional GPFS ACL is associated with the file, a translation to NFS V4 ACL format must be performed when the ACL is read by an NFS V4 client. Since this translation has to be done, an option (**-k nfs4**) is provided on the **mmgetacl** and **mmeditacl** commands, so that this translation can be seen locally as well.

It can also be the case that NFS V4 ACLs have been set for some file system objects (directories and individual files) prior to administrator action to revert back to a POSIX-only configuration. Since the NFS V4 access evaluation will no longer be performed, it is desirable for the **mmgetacl** command to return an ACL representative of the evaluation that will now occur (translating NFS V4 ACLs into traditional POSIX style). The **-k posix** option returns the result of this translation.

Users may need to see ACLs in their true form as well as how they are translated for access evaluations. There are four cases:

1. By default, the **mmgetacl** command returns the ACL in a format consistent with the file system setting:
   - If **posix** only, it is shown as a traditional ACL.
   - If **nfs4** only, it is shown as an NFS V4 ACL.
   - If **all** formats are supported, the ACL is returned in its true form.
2. The command **mmgetacl -k nfs4** always produces an NFS V4 ACL.
3. The command **mmgetacl -k posix** always produces a traditional ACL.

4. The command **mmgetacl -k native** always shows the ACL in its true form, regardless of the file system setting.

In general, users should continue to use the **mmgetacl** and **mmeditacl** commands without the **-k** flag, allowing the ACL to be presented in a form appropriate for the file system setting. Since the NFS V4 ACLs are more complicated and therefore harder to construct initially, users that want to assign an NFS V4 ACL should use the command **mmeditacl -k nfs4** to start with a translation of the current ACL, and then make any necessary modifications to the NFS V4 ACL that is returned.

## Setting NFS V4 access control lists

There is no option on the **mmputacl** command to identify the type (traditional or NFS V4) of ACL that is to be assigned to a file. Instead, the ACL is assumed to be in the traditional format unless the first line of the ACL is:

```
#NFSv4 ACL
```

The lines that follow the first one are then processed according to the rules of the expected ACL type.

An NFS V4 ACL is similar to this:

```
#NFSv4 ACL
#owner:root
#group:system
special:owner@:rwxc:allow
(X)READ/LIST (X)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE (X)READ_ACL  (-)READ_ATTR  (-)READ_NAMED
(X)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (X)WRITE_ATTR (-)WRITE_NAMED

special:owner@:----:deny
(-)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL  (-)READ_ATTR  (X)READ_NAMED
(-)DELETE    (X)DELETE_CHILD (X)CHOWN (-)EXEC/SEARCH (-)WRITE_ACL (-)WRITE_ATTR (X)WRITE_NAMED

user:guest:r-xc:allow
(X)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (X)SYNCHRONIZE (X)READ_ACL  (-)READ_ATTR  (-)READ_NAMED
(X)DELETE    (-)DELETE_CHILD (-)CHOWN (X)EXEC/SEARCH (X)WRITE_ACL (-)WRITE_ATTR (-)WRITE_NAMED

user:guest:----:deny
(-)READ/LIST (-)WRITE/CREATE (-)APPEND/MKDIR (-)SYNCHRONIZE (-)READ_ACL  (-)READ_ATTR  (X)READ_NAMED
(-)DELETE    (X)DELETE_CHILD (X)CHOWN (-)EXEC/SEARCH (-)WRITE_ACL (X)WRITE_ATTR (X)WRITE_NAMED
```

This ACL shows four ACL entries (an allow and deny entry for each of **owner@** and **guest**).

In general, constructing NFS V4 ACLs is more complicated than traditional ACLs. Users new to NFS V4 ACLs may find it useful to start with a traditional ACL and allow either **mmgetacl** or **mmeditacl** to provide the NFS V4 translation, using the **-k nfs4** flag as a starting point when creating an ACL for a new file.

## Displaying NFS V4 access control lists

The **mmgetacl** command displays an existing ACL regardless of its type (traditional or NFS V4). The format of the ACL that is returned depends on the file system setting (**-k** flag), as well as the format of the actual ACL associated with the file. For details, see "NFS V4 ACL translation" on page 251.

## Applying an existing NFS V4 access control list

This function is identical, whether using traditional or NFS V4 ACLs. See "Applying an existing traditional GPFS access control list" on page 247.

## Changing NFS V4 access control lists

This function is identical, whether using traditional or NFS V4 ACLs. See "Changing traditional GPFS access control lists" on page 248.

## Deleting NFS V4 access control lists

Use the **mmdelacl** command to delete NFS V4 ACLs. Once the ACL has been deleted, permissions revert to the mode bits. If the **mmgetacl** command is then used to display the ACL (**mmgetacl -k native**), it appears as a traditional GPFS ACL.

When assigning an ACL to a file that already has an NFS V4 ACL, there are some NFS rules that must be followed. Specifically, in the case of a directory, there will **not** be two separate (access and default) ACLs, as there are with traditional ACLs. NFS V4 requires a single ACL entity and allows individual ACL entries to be flagged if they are to be inherited. Consequently, **mmputacl -d** is not allowed if the existing ACL was the NFS V4 type, since this attempts to change **only** the default ACL. Likewise **mmputacl** (without the **-d** flag) is not allowed because it attempts to change only the access ACL, leaving the default unchanged. To change such an ACL, use the **mmeditacl** command to change the entire ACL as a unit. Alternatively, use the **mmdelacl** command to remove an NFS V4 ACL, followed by the **mmputacl** command.

## Considerations when using GPFS with NFS V4 ACLs

There are several constraints that you need to consider when using GPFS with NFS V4 ACLs. For a complete description of these restrictions, see "GPFS exceptions and limitations to NFS V4 ACLs" on page 276.

---

# NFS and GPFS

GPFS file systems may be exported using the Network File System (NFS) protocol from one or more nodes. After export, normal access to the file system can proceed from GPFS cluster nodes or NFS client nodes.

**Note:** GPFS on Windows does not provide NFS integration.

See the "Authorizing protocol users" on page 258 topic for information on ACLs for protocol users.

Considerations for the interoperability of a GPFS file system include:

- "Exporting a GPFS file system using NFS"
- "NFS usage of GPFS cache" on page 256
- "Synchronous writing using NFS" on page 257
- "Unmounting a file system after NFS export" on page 257
- "NFS automount considerations" on page 257
- "Clustered NFS and GPFS on Linux" on page 257

**Note:** None of these sections take into account the NFS server integration that is introduced with CES. The integrated NFS server interactions, with the following documented sections, will be addressed in a future release.

## Exporting a GPFS file system using NFS

To export a GPFS file system:

1. Create and mount the GPFS file system. In the examples, we assume a file system with a local mount point of **/gpfs**.

   For performance reasons, some NFS implementations cache file information on the client. Some of the information (for example, file state information such as file size and timestamp) is not kept up-to-date in this cache. The client may view stale inode data (on **ls -l**, for example) if exporting a GPFS file system with NFS.

If this is not acceptable for a given installation, caching can be turned off by mounting the file system on the client using the appropriate operating system mount option (for example, **-o noac** on Linux NFS clients). Turning off NFS caching results in extra file systems operations to GPFS, and negatively affect its performance.

**Note:** Ensure that all GPFS file systems that you use to export data via NFS are mounted with the **syncnfs** option to prevent clients from running into data integrity issues during failover. When you mount a GPFS system, it is a good practice to run the **mmchfs** command to set the **syncnfs** option as the default.

2. Make sure that the clocks of all nodes in the GPFS cluster are synchronized. If this is not done, NFS access to the data, as well as other GPFS file system operations, may be disrupted.

   NFS relies on metadata timestamps to validate the local operating system cache. If the same directory is either NFS-exported from more than one node, or is accessed with both the NFS and GPFS mount point, it is critical that clocks on all nodes that access the file system (GPFS nodes and NFS clients) are constantly synchronized using appropriate software (for example, NTP). Failure to do so may result in stale information seen on the NFS clients.

3. Ensure that NFS is properly configured and running.

   For Linux nodes, information on configuring NFS can be obtained at the linuxdocs.org websitewww.linuxdocs.org.

   For AIX nodes, refer to AIX in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/ssw_aix/welcome) for information about configuring NFS.

4. Use the **mmnfs export add**, **mmnfs export change**, or **mmnfs export remove** command as required.

   **Note:** For IBM Spectrum Scale exports, you must use **mmnfs export add**, **mmnfs export change**, or **mmnfs export remove** command. Editing the **/etc/exports** file to specify file systems to be exported does not work for IBM Spectrum Scale exports; this can only be used for exports that are not IBM Spectrum Scale exports.

## Export considerations

Keep these points in mind when exporting a GPFS file system to NFS. The operating system being used and the version of NFS might require special handling or consideration.

**Linux export considerations:** Linux does not allow a file system to be NFS V4 exported unless it supports POSIX ACLs. For more information, see "Linux ACLs and extended attributes" on page 276.

For Linux nodes only, issue the **exportfs -ra** command to initiate a reread of the **/etc/exports** file.

Starting with Linux kernel version 2.6, an **fsid** value must be specified for each GPFS file system that is exported on NFS. For example, the format of the entry in **/etc/exports** for the GPFS directory **/gpfs/dir1** might look like this:

```
/gpfs/dir1 cluster1(rw,fsid=745)
```

The administrator must assign **fsid** values subject to the following conditions:

1. The values must be unique for each file system.
2. The values must not change after reboots. The file system should be unexported before any change is made to an already assigned **fsid**.
3. Entries in the **/etc/exports** file are not necessarily file system roots. You can export multiple directories within a file system. In the case of different directories of the same file system, the **fsid**s must be different. For example, in the GPFS file system **/gpfs**, if two directories are exported (**dir1** and **dir2**), the entries might look like this:

   ```
   /gpfs/dir1 cluster1(rw,fsid=745)
   /gpfs/dir2 cluster1(rw,fsid=746)
   ```

4. If a GPFS file system is exported from multiple nodes, the **fsid**s should be the same on all nodes.

Configuring the directories for export with NFSv4 differs slightly from the previous NFS versions. To configure the directories, do the following:

1. Define the root of the overall exported file system (also referred to as the pseudo root file system) and the pseudo file system tree. For example, to define **/export** as the pseudo root and export **/gpfs/dir1** and **/gpfs/dir2** which are not below **/export**, run:

   ```
   mkdir –m 777 /export /export/dir1 /export/dir2
   mount --bind /gpfs/dir1 /export/dir1
   mount –-bind /gpfs/dir2 /export/dir2
   ```

   In this example, **/gpfs/dir1** and **/gpfs/dir2** are bound to a new name under the pseudo root using the bind option of the **mount** command. These bind mount points should be explicitly umounted after GPFS is stopped and bind-mounted again after GPFS is started. To unmount, use the **umount** command. In the preceding example, run:

   ```
   umount /export/dir1; umount /export/dir2
   ```

2. Edit the **/etc/exports** file. There must be one line for the pseudo root with **fsid=0**. For the preceding example:

   ```
   /export cluster1(rw,fsid=0)
   /export/dir1 cluster1(rw,fsid=745)
   /export/dir2 cluster1(rw,fsid=746)
   ```

   The two exported directories (with their newly bound paths) are entered into the **/etc/exports** file.

Large installations with hundreds of compute nodes and a few login nodes or NFS-exporting nodes require tuning of the GPFS parameters **maxFilesToCache** and **maxStatCache** with the **mmchconfig** command.

The general suggestion is for the compute nodes to set **maxFilesToCache** to about 200. The login or NFS nodes should set this parameter much higher, with **maxFilesToCache** set to 1000 and **maxStatCache** set to 50000.

**Note:** The stat cache is not effective on the Linux platform. Therefore, you need to set the **maxStatCache** attribute to a smaller value, such as 512, and the **maxFilesToCache** attribute to 50000.

This tuning is required for the GPFS token manager (file locking), which can handle approximately 1,000,000 files in memory. The token manager keeps track of a total number of tokens, which equals **5000 * number of nodes**. This will exceed the memory limit of the token manager on large configurations. By default, each node holds 5000 tokens:

- If the user does not specify values for **maxFilesToCache** and **maxStatCache**, the default value of **maxFilesToCache** is 4000, and the default value of **maxStatCache** is 1000.
- On upgrades to GPFS 4.1 from GPFS 3.4 or earlier, the existing defaults (1000 for **maxFilesToCache** and 4000 for **maxStatCache**) remain in effect.

If the user specifies a value for **maxFilesToCache** but does not specify a value for **maxStatCache**, the default value of **maxStatCache** changes to 4***maxFilesToCache**.

If you are running at SLES 9 SP 1, the kernel defines the **sysctl** variable **fs.nfs.use_underlying_lock_ops**, which determines whether the NFS **lockd** is to consult the file system when granting advisory byte-range locks. For distributed file systems like GPFS, this must be set to **true** (the default is **false**).

You can query the current setting by issuing the command:

```
sysctl fs.nfs.use_underlying_lock_ops
```

Alternatively, the **fs.nfs.use_underlying_lock_ops = 1** record can be added to **/etc/sysctl.conf**. This record must be applied after initially booting the node, and after each reboot, by issuing the command:

```
sysctl -p
```

Because the **fs.nfs.use_underlying_lock_ops** variable is currently not available in SLES 9 SP 2 or later, when NFS-exporting a GPFS file system, ensure that your NFS server nodes are at the SP 1 level (unless this variable is made available in later service packs).

For additional considerations when NFS exporting your GPFS file system, refer to *File system creation considerations* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

**AIX export considerations:** AIX does not allow a file system to be exported by NFS V4 unless it supports NFS V4 ACLs.

**NFS export considerations for versions prior to NFS V4:** For NFS exported file systems, the version of NFS you are running with may have an impact on the number of inodes you need to cache, as set by both the **maxStatCache** and **maxFilesToCache** parameters on the **mmchconfig** command. The implementation of the **ls** command differs from NFS V2 to NFS V3. The performance of the **ls** command in NFS V3 in part depends on the caching ability of the underlying file system. Setting the cache large enough will prevent rereading inodes to complete an **ls** command, but will put more of a CPU load on the token manager.

Also, the clocks of all nodes in your GPFS cluster must be synchronized. If this is not done, NFS access to the data, as well as other GPFS file system operations, may be disrupted.

**NFS V4 export considerations:** For information on NFS V4, refer to *NFS Version 4 Protocol* and other information found in the Network File System Version 4 (nfsv4) section of the IETF Datatracker website (datatracker.ietf.org/wg/nfsv4/documents).

To export a GPFS file system using NFS V4, there are two file system settings that must be in effect. These attributes can be queried using the **mmlsfs** command, and set using the **mmcrfs** and **mmchfs** commands.

1. The **-D nfs4** flag is required. Conventional NFS access would not be blocked by concurrent file system reads or writes (this is the POSIX semantic). NFS V4 however, not only allows for its requests to block if conflicting activity is happening, it insists on it. Since this is an NFS V4 specific requirement, it must be set before exporting a file system.

   ```
   flag value          description
   ---- -------------- -------------------------------------------------
    -D nfs4            File locking semantics in effect
   ```

2. The **-k nfs4** or **-k all** flag is required. Initially, a file system has the **-k posix** setting, and only traditional GPFS ACLs are allowed. To export a file system using NFS V4, NFS V4 ACLs must be enabled. Since NFS V4 ACLs are vastly different and affect several characteristics of the file system objects (directories and individual files), they must be explicitly enabled. This is done either exclusively, by specifying **-k nfs4**, or by allowing **all** ACL types to be stored.

   ```
   flag value          description
   ---- -------------- -------------------------------------------------
    -k  all            ACL semantics in effect
   ```

**Note:** In IBM Spectrum Scale 4.2 release, NFS connections are limited to a maximum of 2250 for a large number of NFS exports. The maximum number of NFS exports supported is 1000.

## NFS usage of GPFS cache

Exporting a GPFS file system from a node may result in significant additional demands on the resources at that node. Depending on the number of NFS clients, their demands, and specific mount options, you may want to increase either one or both of the **maxFilesToCache** and **pagepool**. The CES NFS number of open files is set to 1 Mg. By default, gpfs `maxFilesToCache` is set to 4K. If the NFS clients overrun the 4K cache by generating more files, NFS RPC's will fail due to TCP inactivity timeout. Increasing is set to 4K. If the NFS clients overrun the 4K cache by generating more files, NFS RPC's will fail due to TCP inactivity timeout. Increasing `maxFilesToCache`will solve this problem.

See the **mmchconfig** command.

You may also choose to restrict the use of the NFS server node through the normal GPFS path and not use it as either a file system manager node or an NSD server.

## Synchronous writing using NFS

With Linux, **write** operations are usually asynchronous. If synchronous writes are required over NFS, edit the **/etc/exports** file to include **sync,no_wdelay**.

## Unmounting a file system after NFS export

Because NFS use of a GPFS file system may result in a file being held, attempting to unmount a GPFS file system may return a 'Device is busy' error. If this occurs, stop the NFS daemons before attempting to unmount the file system at the NFS server. On Linux, issue this command:

```
/etc/rc.d/init.d/nfs stop
```

On AIX, issue this command:

```
stopsrc -g nfs
```

NFS can be restarted after the unmount completes. On Linux, issue this command:

```
/etc/rc.d/init.d/nfs start
```

On AIX, issue this command:

```
startsrc -g nfs
```

## NFS automount considerations

The default file system type when using the automounter daemon is NFS. When the **-fstype** option is not specified, and the server is the local node, a soft-mount of the local directory is done at the desired mount point. JFS is assumed as the only handler of local directories. A GPFS file system local soft-mount does not work implicitly, since the mount request is passed to JFS which then produces an error. When specifying **-fstype mmfs** the local soft-mount works because the mount is then passed to GPFS instead of JFS.

A GPFS soft-mount does not automatically unmount. Setting **-fstype nfs3** causes the local server mounts to always go through NFS. This allows you to have the same **auto.map** file on all nodes whether the server is local or not, and the automatic unmount will occur. If you want local soft-mounts of GPFS file systems while other nodes perform NFS mounts, you should have different **auto.map** files on the different classes of nodes. This should improve performance on the GPFS nodes as they will not have to go through NFS.

## Clustered NFS and GPFS on Linux

In addition to the traditional exporting of GPFS file systems using NFS, GPFS allows you to configure a subset of the nodes in the cluster to provide a highly available solution for exporting GPFS file systems via NFS.

The participating nodes are designated as Cluster NFS (CNFS) member nodes and the entire setup is frequently referred to as CNFS or CNFS cluster.

In this solution, all CNFS nodes export the same file systems to the NFS clients. When one of the CNFS nodes fails, the NFS serving load moves from the failing node to another node in the CNFS cluster. Failover is done using recovery groups to help choose the preferred node for takeover.

# Authorizing protocol users

Authorization grants or denies access to resources such as directories, files, commands, and functions. Authorization is applicable to an already authenticated identity, such as an IBM Spectrum Scale data user, an administrative user, or an IBM service representative. Access to the files and directories of the IBM Spectrum Scale system is managed through access control lists (ACLs). It ensures that only authorized users get access to exports, directories, and files. An access control entry (ACE) is an individual entry in an access control list, and describes the permissions for an individual user or group of users. An ACL can have zero or more ACEs.

# Authorizing file protocol users

The IBM Spectrum Scale system uses ACLs to authorize users who access the system through file protocols such as NFS and SMB.

The GPFS file system supports storing POSIX and NFSv4 ACLs to authorize file protocol users.

SMB service maps the NFSV4 ACL to a security descriptor for SMB clients to form the ACLs. That is, the SMB ACL is derived from the NFSV4 ACL; it is not a separate ACL. Any changes from SMB clients on ACLs are mapped back to the ACLs in the file system.

To get the expected behavior of ACLs, the file system must be configured to use only the NFSV4 ACLs. The default configuration profiles (`/usr/lpp/mmfs/profiles`) that are included with IBM Spectrum Scale contain the required configuration for NFSV4 ACLs in the file system. When manually creating a file system for protocol usage by using the **mmcrfs** command, use the `-k nfs4` option to establish the correct ACL setting. For more details, see the *mmcrfs command* and the *mmchfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

ACLs can be applied at the following levels:

- Files and directories
- SMB shares

## ACLs on files and directories

The SMB and NFS protocols allow you to manage the ACL permissions on files and directories from connected file systems. ACLs from both protocols are mapped to the same ACL in the file system. The ACL supports inheritance and you can control the inheritance by using the special inheritance flags.

When creating the directory for an SMB or NFS export, the administrator needs to ensure that the ACL on the exported folder is set up as desired. One option is to change the owner of the export folder to an administrator, who can then access the new export and change the ACL from an SMB or NFS client. This is the recommended approach for setting up SMB shares for use with Windows clients.

The recommendation is to manage ACLs from the ACL management interface on a connected client system. For example, after creating the directories for an SMB or NFS export, the owner of this directory can be set with the **chown** command, and this user can connect to the export by using the SMB or NFS protocol to see and manage the ACLs associated with the directory over which the export is created.

## ACLs and POSIX mode bits

The POSIX bits of a file or directory are another authorization method, different from ACLs. POSIX bits can also be used to specify access permissions for a file. You can use the POSIX bits of a file to configure access control for an owner, a group, and for all users to read, update, or run the file. POSIX bits are less flexible than ACLs.

Changing the POSIX modebits also modifies the ACL of an object. When using ACLs for access control, the system administrators might want to ensure that ACLs are not replaced with permissions from POSIX modebits. This behavior can be configured by using the **--allow-permission-change** parameter in **mmcrfileset** and **mmchfileset** commands.

An ACL extends the base permissions or the standard file access modes such as read, write, and execute. ACLs are compatible with UNIX mode bits. Issuing the **chmod** command by the NFS clients overwrite the access privileges that are defined in the ACL by the privileges that are derived from UNIX mode bits. By default, the ACLs are replaced by UNIX mode bits if the **chmod** command is submitted. To allow proper use of ACLs, it is recommended to prevent **chmod** from overwriting the ACLs by setting this parameter to **setAclOnly** or **chmodAndSetAcl**.

NFSV3 clients can set and read the POSIX mode bits; NFSV3 clients who set the UNIX permissions modify the ACL to match the UNIX permissions. In most NFS-only cases, the POSIX permissions are used directly. For NFSV3 clients, file sharing with SMB access protection is done by using NFSV4 ACLs but NFSV3 clients can see only the mapping of ACLs to traditional UNIX access permissions. The full NFSV4 ACLs are enforced on the server.

## SMB protocol export-level ACLs

SMB share ACLs only apply to SMB exports and they are completely separate from the file system ACLs. When using export ACLs, users need to have access in the share-level ACL and in the file or directory.

SMB share ACLs can be changed either through the MMC on a Windows client or through the **mmsmb exportacl** command. For more information, see the **mmsmb exportacl** command on any protocol node.

## ACL inheritance

The inheritance flags in ACL entry of parent directories are used to control the inheritance of authorization to the child files and directories. The inheritance flag gives you the granularity to specify whether the inheritance defined in an ACL entry applies to the current directory and its children or only to the subdirectories and files that are contained in the parent directory. ACL entries are inherited to the child directories or files at the time of creation. Changes made to the ACL of a parent directory are not propagated to child directories or files. However, in case of SMB, you can specify to propagate the inheritance changes from a parent to all its child by using File Explorer, command line, or PowerShell.

### Controlling inheritance of entries inside an ACL

The NFSV4 protocol uses the following flags to specify and control inheritance information of the ACEs:

- *FileInherit:* Indicates that this ACE must be added to each new non-directory file created. This flag is signified by 'f' or file_inherit.
- *DirInherit:* Indicates that this ACE must be added to each new directory created. This flag is signified by `'d'` or `dir_inherit`.
- *InheritOnly:* Indicates that this ACE is not applied to the parent directory itself, but only inherited by its children. This flag is signified by 'i' or inherit_only.
- *NoPropagateInherit:* Indicates that the ACL entry must be included in the initial ACL for subdirectories that are created in this directory but not further propagated to subdirectories created below that level.

In case of SMB, the following list shows how the inheritance flags are linked to the Microsoft Windows inheritance modes:

- This folder only (No bits)
- This folder, subfolder, and files (FileInherit, DirInherit)
- This folder and subfolders (DirInherit)
- This folder and files (FileInherit)
- Subfolders and files only (FileInerhit, DirInherit, InheritOnly)

- Subfolders only (DirInherit, InheritOnly)
- Files only (FileInherit, InheritOnly)

## ACL best practices

It is essential to properly apply ACLs to the file systems, filesets, and exports, and directories and files to ensure smooth access for the users.

Consider the following points before you create or copy data into the export:

1. Should a group of users be given permission to access the data?
2. Should individual users be given permission to access the data?
3. Should selected users from different groups be given access to selected data?
4. Should the shares be in mixed mode? That is, do you have NFS and SMB clients who access the exports in explicit mode, where the data is accessed either from SMB or NFS?
5. Should the applications that the clients are using over SMB and NFS be given any specific ACL permission?

### Setting ACLs for groups

The recommended way to manage access is per group instead of per individual user. This way, users can be easily added to or removed from the group. Providing ACLs to groups has an added advantage of managing inheritance easily for the whole group of users simultaneously. If individual users are added directly to ACLs and you need to make a change, you need to update ACLs of all corresponding directories and files. On the authentication server like Active Directory or LDAP, you can create groups and add users as members and use these groups to give respective access to data.

### Setting ACLs for individual users

If you need to set ACLs for individual users where data is created by users in folders that are created by others, it is recommended that you explicitly add the users who need ACLs on that export.

In mixed mode, where the share is used for both NFS and SMB access, parent owner might experience loss of access to the child directory or the files. To avoid such a problem, it is recommended that you provide ACLs explicitly to each user.

### Special Owner and Group

The special owner and group dynamically refer to the owner and group of the directory or file that the ACL belongs to. For example, if the owner of a file is changed, all special:owner@ entries in the ACL refers to the new owner. In case of inheritance, this leads to some complexity because those special entries point to the owner and group of the child directory or file that inherits the entry. In many cases, the children do not have the same owner and group as the parent directory. Therefore, the special entries in parent and children refer to different users. This can be avoided by adding static entries (user:'name' or group:'name') to the ACL. These static entries are inherited by name and refer everywhere to the same users. But they are not updated if the owner of the parent is changed. The general recommendation is not to use special:owner@ and special:group@ together with inheritance flags. For more information, see the **mmputacl** command.

### Setting ACLs for special IDs

The inheritance of ACL from the owner of a directory to subdirectories and files works only for subdirectories and files that have the same owner as the parent directory. A subdirectory or file that is created by a different owner does not inherit the ACL of a parent directory that is owned by another user.

In case of special access to NFSV4 exports, parent owners might experience loss of access to its child folders and files. To avoid such a problem, for mixed mode, it is recommended that you provide ACLs to groups rather than to individual users.

## ACL permissions that are required to work on files and directories

The ACL permissions such as Read Permissions and Read Attributes are required to list a file. A file owner requires only the Read Attributes permission to list a file, since the permission Read Permissions is implied. A different user needs to have both the Read Permissions and Read Attributes permissions enabled to reliably list the file. These permissions are both automatically granted together when read access is granted. If the file is already in the cache of the system because it was listed recently, any user is able to list the file, regardless of the values of the Read Permissions and Read Attributes permission values.

The following table describes the ACL permissions that are required when the user of the file is not the file owner, where "X" denotes permission that is required on file or directory and "P" denotes permission that is required on the parent directory of the file or directory.

*Table 28. ACL permissions required to work on files and directories, while using SMB protocol (table 1 of 2)*

| ACL Operation | ACL Permission | | | | | |
|---|---|---|---|---|---|---|
| | **Traverse folder / execute file** | **List folder / read data** | **Read attribute** | **Read extended attribute** | **Create files / write data** | **Create folders / append data** |
| Execute file | X | X | | | | |
| List folder | | X | | | | |
| Read data from file | | X | X | X | | |
| Read attributes | | | X | | | |
| Create file | | | | | X | |
| Create folder | | | | | | X |
| Write data to file | | X | X | | X | X |
| Write file attributes | | | | | | |
| Write folder attributes | | | | | | |
| Delete file | | P | X | | P | |
| Delete folder | | P | X | | P | |
| Rename file | | P | X | | P | |
| Rename folder | | P | X | | P | P |
| Read file permissions | | | | | | |
| Read folder permissions | | | | | | |
| Write file permissions | | | | | | |
| Write folder permissions | | | | | | |
| Take file ownership | | | | | | |
| Take folder ownership | | | | | | |

*Table 29. ACL permissions required to work on files and directories, while using SMB protocol (table 2 of 2)*

| ACL Operation | ACL Permission | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Write attribute** | **Write extended attributes** | **Delete subfolder and files** | **Delete** | **Read permissions** | **Write permissions** | **Take ownership** |
| Execute file | | | | | | | |
| List folder | | | | | | | |
| Read data from file | | | | | | | |
| Read attributes | | | | | | | |
| Create file | | | | | | | |
| Create folder | | | | | | | |
| Write data to file | X | X | | | | | |
| Write file attributes | X | | | | | | |
| Write folder attributes | X | | | | | | |
| Delete file | | | P or X | | | | |
| Delete folder | | | P or X | | | | |
| Rename file | | | P or X | | | | |
| Rename folder | | | P or X | | | | |
| Read file permissions | | | | | X | | |
| Read folder permissions | | | | | X | | |
| Write file permissions | | | | | X | X | |
| Write folder permissions | | | | | X | X | |
| Take file ownership | | | | | | | X |
| Take folder ownership | | | | | | | X |

*Table 30. ACL permissions required to work on files and directories, while using NFS protocol (table 1 of 2)*

| ACL Operation | ACL Permission | | | | | |
|---|---|---|---|---|---|---|
| | **Traverse folder / execute file** | **List folder / read data** | **Read attribute** | **Read extended attribute** | **Create files / write data** | **Create folders / append data** |
| Execute file | P, X | X | | | | |
| List folder | P | X | | | | |
| Read data from file | P | X | | | | |
| Read attributes | P | | | | | |
| Create file | P | | | | P | |
| Create folder | P | | | | | P |
| Write data to file | P | | | | X | X |

*Table 30. ACL permissions required to work on files and directories, while using NFS protocol (table 1 of 2) (continued)*

| ACL Operation | ACL Permission | | | | | |
|---|---|---|---|---|---|---|
| | **Traverse folder / execute file** | **List folder / read data** | **Read attribute** | **Read extended attribute** | **Create files / write data** | **Create folders / append data** |
| Write file attributes | P | | | | | |
| Write folder attributes | P | | | | | |
| Delete file | P | | | | P | |
| Delete folder | P | | | | P | |
| Rename file | P | | X | | P | |
| Rename folder | P | | X | | P | P |
| Read file ACL | P | | | | | |
| Read folder ACL | P | | | | | |
| Write file ACL | P | | | | | |
| Write folder ACL | P | | | | | |
| Take file ownership | P | | | | | |
| Take folder ownership | P | | | | | |

*Table 31. ACL permissions required to work on files and directories, while using NFS protocol (table 2 of 2)*

| ACL Operation | ACL Permission | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Write attribute** | **Write extended attributes** | **Delete subfolder and files** | **Delete** | **Read ACL** | **Write ACL** | **Take ownership** |
| Execute file | | | | | | | |
| List folder | | | | | | | |
| Read data from file | | | | | | | |
| Read attributes | | | | | | | |
| Create file | | | | | | | |
| Create folder | | | | | | | |
| Write data to file | | | | | | | |
| Write file attributes | | | | | | | |
| Write folder attributes | | | | | | | |
| Delete file | | | P | | | | |
| Delete folder | | | P | | | | |
| Rename file | | | P | | | | |
| Rename folder | | | P | | | | |
| Read file ACL | | | | | | | |
| Read folder ACL | | | | | | | |
| Write file ACL | | | | | | X | |
| Write folder ACL | | | | | | X | |
| Take file ownership | | | | | | | X |

*Table 31. ACL permissions required to work on files and directories, while using NFS protocol (table 2 of 2) (continued)*

| ACL Operation | ACL Permission | | | | | | |
|---|---|---|---|---|---|---|---|
| | **Write attribute** | **Write extended attributes** | **Delete subfolder and files** | **Delete** | **Read ACL** | **Write ACL** | **Take ownership** |
| Take folder ownership | | | | | | | X |

The following are the considerations on the ACL read and write permissions:

1. For the "Read data from file" operation, the IBM Spectrum Scale system checks the validity of the client requested access mask only if "Read permissions" attribute is enabled on the file. If "Read permissions" attribute is not enabled, then only the "List folder / read data" and "Read attributes" permissions are required to read from the file.
2. For the "Write data to file" operation, the IBM Spectrum Scale system checks the validity of the client requested access mask only if the "Read permissions" attribute is enabled on the file. If the "Read permissions" attribute is not enabled, then only the "Create files / write data" and "Create folders / append data" permissions are required to write to the file.
3. The files that require "Traverse folder / execute file" permission do not require the "Bypass Traverse Check" attribute to be enabled. This attribute is enabled by default on the files.
4. The "Read extended attribute" permission is required by the SMB clients with recent Microsoft Windows versions (for Microsoft Windows 2008, Microsoft Windows 2012, and Microsoft Windows 8 versions) for file copy operations. The default ACLs set without inheritance do not contain this permission. It is recommended that you use inherited permissions where possible and enable this permission in the inherited permissions to prevent the default value to be used and cause problems.

Migrating data through SMB to the IBM Spectrum Scale cluster requires a user ID with the enhanced permissions. The ownership of a file cannot be migrated by a normal IBM Spectrum Scale user. Therefore, you need to configure an “admin user” to allow data migration. For more information on how to configure the “admin users” parameter, see the *mmsmb export add* and *mmsmb export change* sections in *mmsmb command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Directory traversal permissions that are applicable for SMB ACLs

The following are the considerations on the traverse permissions:

1. It is recommended that you add the "Traverse folder / execute file" permission to all executable files, even if the "Bypass Traverse Check" attribute is enabled on these files. IBM Spectrum Scale checks for the "Traverse folder / execute file" permission on executable files irrespective of the value of the "Bypass Traverse Check" attribute.
2. If the `--cifsBypassTraversalChecking` option is enabled, it allows a user to directly access files and folders that the user owns, and also that are contained under the parent folders for which the user does not have Read or Write permissions. Users without "Read and Execute" access to the share or export in which the user-owned files and folders are located can read and modify the files inside the export for which the user has permissions that are granted by the `--cifsBypassTraversalChecking` option. However, in this case, operations like rename file and delete file are not granted by default. This is normal SMB behavior. Modify ACLs as required to enable these operations.

   For example, in the directory structure /A/B/C, assume that an SMB user has 'read' permission on C but no permissions on A and B. When the `--cifsBypassTraversalChecking` option is set to its default value Yes, this SMB user can access C without having "Traverse Folder" or "Execute File" permissions that are set to allow on A and B, but is still not allowed to browse the content of A and B.
3. The ownership of a file cannot be migrated by a normal user. You must configure and use administrative user credentials to perform data migration. When migrating existing files and directories from other systems to IBM Spectrum Scale, the ACL might not contain explicit traversal

rights for the users because the source system can grant this right implicitly. After migrating the files with ACLs, ensure that traversal rights are granted to the parent directory of each exported path.

## Working with ACLs

The IBM Spectrum Scale system applies default ACLs for newly created IBM Spectrum Scale file system components such as file system, filesets, file, directories, and exports.

The file system must be created with native ACL type as NFS V4. It is recommended to use the default configuration profiles (`/usr/lpp/mmfs/profiles`) that are included with IBM Spectrum Scale. It contains the required configuration for NFSV4 ACLs in the file system.

### Applying default ACLs

Perform the following steps to apply default ACLs on SMB and NFS exports:

1. Create a fileset or directory in the file system as shown in the following example:
   ```
   mkdir -p /ibm/gpfs0/testsmbexport
   ```
2. Change the owner and group of the fileset or directory using **chown** and **chgrp** respectively. For example:
   ```
   chown -R "DOMAIN\\username":"DOMAIN\\groupname" /ibm/gpfs0/testsmbexport
   ```
3. Use the **mmputacl** or **mmeditacl** commands to set the wanted ACE along with specific ACE for owner user and owner group and inheritance flags for the fileset or directory.
4. Check the ACL setting for the fileset or directory by using the **mmgetacl** command.
5. Create the desired SMB or NFS export by using the **mmnfs** or **mmsmb** commands over the fileset or directory.
6. For data exported for SMB clients, it is recommend to manage the ACLs from a Windows clients, since there is already a GUI interface available and the ACL is set according to the requirements of Windows clients. Modifying the ACL directly with **mmputacl** and **mmeditacl** are not advised.

### Viewing the owner of the SMB share

Perform the following steps to create an SMB share and view the owner of the export:

1. Submit the **mmsmb export add** command to create an SMB share as shown in the following example:
   ```
   mmsmb export add testsmbexport /ibm/gpfs0/testsmbexport
   ```
2. Issue either the **ls -la** command or the **mmgetacl** command to view the owner of the export. For example:
   ```
   ls -la /ibm/gpfs0/testsmbexport
   ```

   Or
   ```
   mmgetacl /ibm/gpfs0/testsmbexport
   ```

Apart from the tasks that are listed earlier in this section, the following table provides a quick overview of the tasks that can be performed to manage ACLs and the corresponding IBM Spectrum Scale command.

*Table 32. Commands and reference to manage ACL tasks.*

| Tasks that can be performed to manage ACLs | Command | Reference topic |
|---|---|---|
| Applying ACL at file system, fileset, and export level | **mmeditacl** | "Applying an existing NFS V4 access control list" on page 252 |
| Inserting ACEs in existing ACLs | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |

*Table 32. Commands and reference to manage ACL tasks (continued).*

| Tasks that can be performed to manage ACLs | Command | Reference topic |
|---|---|---|
| Modifying ACLs | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Copying Access control list entries | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Replacing a complete ACL | **mmputacl** or **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Replacing all entries for a specific user inside an ACL | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Controlling inheritance of entries inside an ACL | **mmputacl** or **mmeditacl** | |
| Deleting complete ACL | **mmdelacl** | "Deleting NFS V4 access control lists" on page 253 |
| Deleting specific ACL entries | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Deleting ACL entry for a user | **mmeditacl** | "Changing NFS V4 access control lists" on page 252 |
| Displaying an ACL | **mmgetacl** | "Displaying NFS V4 access control lists" on page 252 |
| Changing file system directory's owner and group | **chown** or **chgroup** | |
| Displaying file system directory's owner and group | **ls –l** or **mmgetacl** | |

**Important:** The **mmgetacl**, **mmputacl**, and **mmeditacl** commands are available to change the ACLs directly. As the SMB clients might depend on the order of entries in the ACL, it is not recommended to change the ACLs directly on GPFS while using the SMB protocol. Changing an ACL directly in GPFS also does not account for inherited entries. So, it is recommended to change the ACLs from a windows client.

### Managing ACLs from Windows clients

For SMB shares, it is recommended to manage the ACLs from a Windows client. The following operations are included in creating an SMB share:

1. Create the folder to export in the file system with the mkdir command.
2. Change the owner of the exported folder to a user who configures the initial ACLs.
3. Create the export using the **mmsmb export create** command.
4. Using a Windows client machine, access the newly created share as the user specified in step 2.
5. Right-click on the shared folder, and select **Properties**.
6. Select the **Security** tab and then select **Advanced** to navigate to the more detailed view of permissions.
7. Add and remove permissions as required.

## Authorizing object users

The Object Storage service of the IBM Spectrum Scale system uses Keystone service for Identity Management. The Identity Management service consists of user authentication and authorization processes.

Access for the object users to the Object Storage projects are controlled by the user roles and container ACLs. Based on the roles defined for the user, object users can be administrative users and non-administrative users. Non-admin users can only perform operations per container based on the container's X-Container-Read and X-Container-Write ACLs. Container ACLs can be defined to limit access to objects in swift containers. Read access can be limited to only allow download, or allow download and listing. Write access allows the user to upload new objects to a container.

You can use an external AD or LDAP server or a local database as the back-end to store and manage user credentials for user authentication. The authorization details such as relation of users with projects and roles are maintained locally by the keystone server. The customer can select the authentication server to be used. For example, if AD is existing in an enterprise deployment and the users in AD are required to access object data, the customer can decide to use AD as the back-end authentication server.

When the back-end authentication server is AD or LDAP, the user management operations such as creating user, deleting user, and so on are the responsibility of the AD or LDAP administrator, who can optionally also be the Keystone server administrator. When local authentication is used for object access, the user management operations are done by the Keystone administrator. In case of authorization, the management tasks such as creating roles, projects, and associating the user with them is done by the Keystone Administrator. The Keystone administration can be done through the Keystone V3 REST API or by using an OpenStack python-based client.

Before you start creating object users, and projects, ensure that Keystone server is configured and the authentication servers are set up properly. You can use the **mmces service list -a -v** command to see whether Keystone is configured properly.

The object users are authorized to the object data and resources by creating and managing roles and ACLs. The roles and ACLs define the actions that can be performed by the user on the object resources such as accessing data, managing the projects, creating projects, read, write, run permissions, and so on.

## Configuring container ACLs to authorize object data users

The following examples and sections provide an understanding on how to set up container ACLs and define the access permissions for the user.

**Creating containers:**

The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

It is the responsibility of the Keystone server administrator to create and manage accounts. The account defines a namespace for containers. A container must be unique within the owning account and account must use a unique name within the project. The `admin` account is created by default.

The following is an example of how to create containers.

**GUI navigation**

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

1. Issue the **swift post container** command to create a container by using the Swift Command Line Client. In the following example, the Keystone administrator creates a `public_readOnly` container in `admin` account.

```
# swift post public_readOnly  --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
```

2. Issue the **swift list** command to list the containers that are available for the account. In the following example, the system lists the containers that are available in the admin project.

```
# swift list --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
public_readOnly
```

3. Issue the **swift stat** command to list the accounts, containers, or objects details. In the following example, the system displays the admin account details.

```
# swift stat -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
                     StorageURL: http://tully-ces-ip.adcons.spectrum:8080/v1
/AUTH_bea5a0c632e54eaf85e9150a16c443ce
                     Auth Token: 1f6260c4f8994581a465b8225075c932
                        Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
                     Containers: 1
                        Objects: 0
                          Bytes: 0
Containers in policy "policy-0": 1
Objects in policy "policy-0": 0
     Bytes in policy "policy-0": 0
    X-Account-Project-Domain-Id: default
                   X-Timestamp: 1432766053.43581
                    X-Trans-Id: tx9b96c4a8622c40b3ac69a-0055677ce7
                  Content-Type: text/plain; charset=utf-8
                 Accept-Ranges: bytes
```

In the following example, the system displays the public_readOnly' container details, on the admin account:

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly
      Auth Token: 957d6c37155b44d3a476441bc927835d
         Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
       Container: public_readOnly
         Objects: 0
           Bytes: 0
        Read ACL:
       Write ACL:
         Sync To:
        Sync Key:
   Accept-Ranges: bytes
X-Storage-Policy: Policy-0
     X-Timestamp: 1432795292.10297
      X-Trans-Id: tx9b05c2135a9c4034b910c-0055677dad
    Content-Type: text/plain; charset=utf-8
```

By default, only users who are having a Keystone role specified in the proxy-server.conf operator_roles option are allowed to create container on an account.

To list operators_role on the IBM Spectrum Scale system during installation, issue the **mmobj config list** command as shown in the following example:

```
mmobj config list --ccrfile proxy-server.conf --section filter:keystoneauth --property operator_roles
```

To list operators_role in all other cases, issue the **mmobj config list** with the following parameters:

```
mmobj config list --ccrfile proxy-server.conf --section filter:keystone --property operator_roles
```

Keystone administrator can also use the container to control access to the objects by using an access control list (ACL). The following example shows that when a member of the admin account tries to display the details of public_readOnly account, the process fails because it does not have an operator role or access control defined:

```
# swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username member
--os-user-domain-name Default --os-password Passw0rd --auth-version 3
Container HEAD failed: http://tully-ces-ip.adcons.spectrum:8080/v1
/AUTH_bea5a0c632e54eaf85e9150a16c443ce/public_readOnly 403 Forbidden
```

**Related tasks**:

"Creating read ACLs to authorize object users"
The Keystone administrator can create container ACLs to grant read permissions using `X-Container-Read headers` in curl tool or `–read-acl flag` in the Swift Command Line Client.

"Creating write ACLs to authorize object users" on page 271
The Keystone administrator can create container ACLs to grant write permissions using X-Container-Write headers in the curl tool or `–write-acl flag` in the Swift Command Line Client.

**Creating read ACLs to authorize object users:**

The Keystone administrator can create container ACLs to grant read permissions using `X-Container-Read headers` in curl tool or `–read-acl flag` in the Swift Command Line Client.

The following example shows how to create read permission in an ACL.

1. Upload the object *imageA.JPG* to *public_readOnly* container as the Keystone administrator.

   ```
   # swift upload public_readOnly imageA.JPG --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
   --os-project-name admin --os-project-domain-name Default --os-username admin
   --os-user-domain-name Default --os-password Passw0rd --auth-version 3
   imageA.JPG
   ```

2. Issue the **swift post** command to provide public read access to the *public_readOnly* container.

   ```
   # swift post public_readOnly --read-acl ".r:*,.rlistings" --os-auth-url http://tully-ces-ip.
   adcons.spectrum:35357/v3 --os-project-name admin --os-project-domain-name Default
   --os-username admin --os-user-domain-name Default --os-password Passw0rd --auth-version 3
   ```

   **Note:** The `.r:*` ACL specifies access for any referrer regardless of account affiliation or user name. The `.rlistings` ACL allows to list the containers and read (download) objects.

3. Issue the **swift stat** command at the container level to see the access details.

   ```
   # swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
   --os-project-name admin --os-project-domain-name Default --os-username admin
   --os-user-domain-name Default --os-password Passw0rd --auth-version 3
               URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
   /public_readOnly
         Auth Token: 91a27a5ed8dc40d582e71844ca019c32
            Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
          Container: public_readOnly
            Objects: 3
              Bytes: 8167789
           Read ACL: .r:*,.rlistings
          Write ACL:
            Sync To:
           Sync Key:
      Accept-Ranges: bytes
         X-Trans-Id: tx73b0696705b94bf885bd5-0055678ab1
   X-Storage-Policy: Policy-0
        X-Timestamp: 1432795292.10297
       Content-Type: text/plain; charset=utf-8
   ```

4. As the *student* user from the *students* account, list and download the details of *public_readOnly* container that is created in the *admin* account.

   Listing the details:

   ```
   # swift stat public_readOnly -v --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
   --os-project-name students --os-project-domain-name Default --os-username student1
   --os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
   http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
               URL: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
   ```

```
/public_readOnly
      Auth Token: d6ee0fb5e33748b1b9035a3b690c7587
         Account: AUTH_bea5a0c632e54eaf85e9150a16c443ce
       Container: public_readOnly
         Objects: 3
           Bytes: 8167789
        Read ACL:
       Write ACL:
         Sync To:
        Sync Key:
   Accept-Ranges: bytes
X-Storage-Policy: Policy-0
     X-Timestamp: 1432795292.10297
      X-Trans-Id: tx09893920a6154faab6ace-0055678f6d
    Content-Type: text/plain; charset=utf-8
```

Listing the container objects:

```
# swift list public_readOnly --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
file.txt
imageA.JPG
imageB.JPG
```

Downloading container objects:

```
# swift download public_readOnly --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
imageB.JPG [auth 0.321s, headers 0.380s, total 0.390s, 37.742 MB/s]
file.txt [auth 0.533s, headers 0.594s, total 0.594s, 0.000 MB/s]
imageA.JPG [auth 0.119s, headers 0.179s, total 18.135s, 0.308 MB/s]
```

5. As the *student1* user from the *students* account, receive deny write access, while trying to upload new content in the *public_readOnly* container:

```
# swift upload public_readOnly photo.jpg --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --auth-version 3 --os-storage-url
http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
Warning: failed to create container 'public_readOnly': 403 Forbidden:

Forbidden

Access was denied to this resourc
Object PUT failed: http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/public_readOnly/photo.jpg 403 Forbidden
```

### Manipulating the read ACLs

The following table list different read ACLs combinations:

*Table 33. ACL options that are available to manipulate object read ACLs*

| Permission | Read ACL options |
|---|---|
| Read for all referrers | .r:* |
| Read and list for all referrers and listing | .r:*,.rlistings |
| Read and list for a user in a specific project | <project_name\|project_id>:<user_name\|user_id> |
| Read and list for a user in every project | *:<user_name\|user_id> |
| Read and list for every user in a project | <project_name\|project_id>:<*> |
| Read and list for every user in every project | <*>:<*> |

**Note:** Use comma (,) to separate ACLs. For example, `–read-acl admin:admin,students:student1`.

**Related tasks**:

"Creating containers" on page 267
The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

"Creating write ACLs to authorize object users"
The Keystone administrator can create container ACLs to grant write permissions using X-Container-Write headers in the curl tool or `–write-acl flag` in the Swift Command Line Client.

**Creating write ACLs to authorize object users:**

The Keystone administrator can create container ACLs to grant write permissions using X-Container-Write headers in the curl tool or `–write-acl flag` in the Swift Command Line Client.

Provides an example on how to configure write ACLs by using curl tool.

1. Create token and proceed to create a container named *writeOnly* with write permissions for *member* user who is part of the *admin* project and *student1* user who is part of the *students* project.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X PUT -H "Content-Length: 0" -H "X-Auth-Token: ${token}" -H
"X-Container-Write: admin:member,students:student1" -H "X-Container-Read: "
HTTP/1.1 201 Created
Content-Length: 0
Content-Type: text/html; charset=UTF-8
X-Trans-Id: txf7b0bfef877345949c61c-005567b9d1
Date: Fri, 29 May 2015 00:58:57 GMT
```

2. Issue a token as *student1* from the *students* project and upload an object by using the curl tool.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly/imageA.JPG -X PUT -H "X-Auth-Token: ${token}"  --upload-file  imageA.JPG
HTTP/1.1 100 Continue
HTTP/1.1 201 Created
Last-Modified: Fri, 29 May 2015 01:11:28 GMT
Content-Length: 0
Etag: 95d8c44b757f5b0c111750694dffef2b
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx6caa0570bfcd419782274-005567bcbe
Date: Fri, 29 May 2015 01:11:28 GMT
```

3. List the state of the *writeOnly* container as *student1* user of the *students* project. This operation fails as the user does not have the required privileges.

```
# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly/imageA.JPG -X HEAD  -H "X-Auth-Token: ${token}"
HTTP/1.1 403 Forbidden
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx4f7dfbfd74204785b6b50-005567bd8c
Content-Length: 0
Date: Fri, 29 May 2015 01:14:52 GMT
```

4. Grant read permissions to *student1* user of the *students* project:

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name admin --os-project-domain-name Default --os-username admin
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_
bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X POST -H "Content-Length: 0" -H "X-Auth-Token:
${token}" -H "X-Container-Read: students:student1"
HTTP/1.1 204 No Content
Content-Length: 0
Content-Type: text/html; charset=UTF-8
X-Trans-Id: tx77aafe0184da4b68a7756-005567beac
Date: Fri, 29 May 2015 01:19:40 GMT
```

5. Verify whether the *sutdent1* user has the read access now.

```
token=$(openstack --os-auth-url http://tully-ces-ip.adcons.spectrum:35357/v3
--os-project-name students --os-project-domain-name Default --os-username student1
--os-user-domain-name Default --os-password Passw0rd --os-identity-api-version 3
token issue | grep -w "id" | awk '{print $4}')

# curl -i http://tully-ces-ip.adcons.spectrum:8080/v1/AUTH_bea5a0c632e54eaf85e9150a16c443ce
/writeOnly -X GET  -H "X-Auth-Token: ${token}"
HTTP/1.1 200 OK
Content-Length: 11
X-Container-Object-Count: 1
Accept-Ranges: bytes
X-Storage-Policy: Policy-0
X-Container-Bytes-Used: 5552466
X-Timestamp: 1432861137.91693
Content-Type: text/plain; charset=utf-8
X-Trans-Id: tx246b39018a5c4bcb90c7f-005567bff3
Date: Fri, 29 May 2015 01:25:07 GMT

imageA.JPG
```

**Note:** Object Storage does not support public write ACLs.

**Related tasks**:

"Creating containers" on page 267
The Object Storage organizes data in account, container, and object. Each account and container is an individual database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

"Creating read ACLs to authorize object users" on page 269
The Keystone administrator can create container ACLs to grant read permissions using `X-Container-Read headers` in curl tool or `–read-acl flag` in the Swift Command Line Client.

# Authorization limitations

Authorization limitations are specific to the protocols that are used to access data.

## NFS ACL limitations

ACLs are stored as NFSv4 ACLs in the file system.

For more information on the known limitations of the NFSV4 ACLs, see "GPFS exceptions and limitations to NFS V4 ACLs" on page 276.

## SMB ACL limitations

The following are the SMB ACL limitations:

- ACL of a new child file or directory depends on the ACL type, the file system settings, and the ACL of the parent directory. Depending on these variables, the results in the IBM Spectrum Scale might be slightly different than in Microsoft Windows. For example, if the parent directory is set to have two ACEs, for example full access for owner and for everyone, the Windows default is to create two ACLs for the child; one is to allow full access for owner and other to allow full access for everyone. The IBM Spectrum Scale system by default creates six ACLs to allow and deny ACLs for owner, group, and everyone.
- If domain server manages the UID and GID mapping, the UID and GID mappings must be configured in the domain server before an ACE for that user or group can be created.
- Users and groups that belonged to another domain, and was migrated to a new domain by using the SID-History mechanism, cannot be stored in an ACL.
- Most well-known SIDs and built-in SIDs cannot be stored in an ACL. Only the "Everyone" SID can be stored and used in an IBM Spectrum Scale system.
- The SMB ACLs cannot be modified when LDAP-based authentication is used for file access.
- Microsoft Windows enables you to limit the scope of inheritance for an ACE to one inheritance by selecting the **Apply these permissions to objects and/or containers within this container only** check box in the Windows Explorer. The IBM Spectrum Scale system does not support to configure this option and limit the scope of inheritance for an ACL.
- ACL inheritance stops at fileset junction points; new filesets always have the default ACL (770 root root).
- The root path of every SMB share needs read permission (read data, read attribute, read extended attribute) for everyone, to prevent the unexpected behavior of, for example, Windows Explorer.
- To prevent display of Access Denied errors, the user must have the read attribute permission on all parent directories, when they have access to a file or directory.
- The value of the dacl_protected bit related to the Include Inheritable permissions from this object's parent check box can be changed only through SMB. The ACL commands cannot access this field. Setting a new ACL resets this field.
- The commands that are used to work on the ACLs do not support recursive updates of inherited ACEs in the file tree.
- Access privileges defined in Windows are not honored. Those privileges are tied to administrator groups and allow access, where the ACL alone does not grant it.
- Audit and alarm ACEs are not supported inside an ACL.
- The Bypass Traverse Check is implemented in GPFS for SMB clients only. Clients that use other protocols might still be locked out because the parent tree of an export has more restrictive ACLs than the export itself.
- POSIX-style ACLs are not supported.
- Similar to the POSIX standard, to read the content of a subdirectory, apart from the read permission in the ACL of this subdirectory, the user also need to have traversal permission (SEARCH in Windows, EXECUTE in POSIX) for all of the parent directories. You can set the traverse permission in the “Everyone” group ACE at the share root, and inherit this privilege to all subdirectories. For the SMB protocol, this is applicable only if the configuration option *bypassTraversalCheck* is disabled.
- Even though the underlying file system does not enforce the permissions for extended attributes (READ_NAMED and WRITE_NAMED), these are enforced for SMB clients.

## ACL limitations that are applicable to all protocols

The following limitations are applicable to all protocols:

- When creating a file system, the user needs to specify -k nfs4 to specifically use NFSv4 ACLs, otherwise the default -k all uses both POSIX ACLs and NFSV4 ACLs.
- The IBM Spectrum Scale Object Storage does not do file share with NFS and SMB.

# Chapter 20. Considerations for GPFS applications

Application design must consider the exceptions to the Open Group technical standards for the **stat()** system call and NFS V4 ACLs. Also, a technique to check if a file system is controlled by GPFS has been provided.

Consider the following:

- "Exceptions to Open Group technical standards"
- "Determining if a file system is controlled by GPFS"
- "GPFS exceptions and limitations to NFS V4 ACLs" on page 276

## Exceptions to Open Group technical standards

GPFS is designed in a way that most applications written to the Open Group technical standard for file system calls can access the GPFS data without any modification. However, there are some exceptions.

Applications that depend on exact reporting of changes to the following fields returned by the **stat()** call may not work as expected:

1. **exact mtime**
2. **mtime**
3. **ctime**
4. **atime**

Providing exact support for these fields would require significant performance degradation to all applications executing on the system. These fields are guaranteed accurate when the file is closed.

These values will be accurate on a node right after it accesses or modifies a file, but may not be accurate for a short while when a file is accessed or modified on some other node.

If 'exact mtime' is specified for a file system (using the **mmcrfs** or **mmchfs** commands with the **-E yes** flag), the **mtime** and **ctime** values are always correct by the time the **stat()** call gives its answer. If 'exact mtime' is not specified, these values will be accurate after a couple of minutes, to allow the synchronization daemons to propagate the values to all nodes. Regardless of whether 'exact mtime' is specified, the **atime** value will be accurate after a couple of minutes, to allow for all the synchronization daemons to propagate changes.

Alternatively, you may use the GPFS calls, **gpfs_stat()** and **gpfs_fstat()** to return exact **mtime** and **atime** values.

The delayed update of the information returned by the **stat()** call also impacts system commands which display disk usage, such as **du** or **df**. The data reported by such commands may not reflect changes that have occurred since the last sync of the file system. For a parallel file system, a sync does not occur until all nodes have individually synchronized their data. On a system with no activity, the correct values will be displayed after the sync daemon has run on all nodes.

## Determining if a file system is controlled by GPFS

A file system is controlled by GPFS if the **f_type** field in the **statfs** structure returned from a **statfs()** or **fstatfs()** call has the value 0x47504653, which is GPFS in ASCII.

This constant is in the **gpfs.h** file, with the name **GPFS_SUPER_MAGIC**. If an application includes **gpfs.h**, it can compare **f_type** to **GPFS_SUPER_MAGIC** to determine if the file system is controlled by GPFS.

## GPFS exceptions and limitations to NFS V4 ACLs

GPFS has some exceptions and limitations for the NFS V4 ACLs.

Those exceptions and limitations include:

1. Alarm type ACL entries are not supported.
2. Audit type ACL entries are not supported.
3. Some types of access for which NFS V4 defines controls do not currently exist in GPFS. For these, ACL entries will be accepted and saved, but since there is no corresponding operation they will have no effect. These include **READ_NAMED**, **WRITE_NAMED**, and **SYNCHRONIZE**.

   **Note:** Even if GPFS ignores these bits, the SMB service will enforce them on the protocol level.
4. AIX requires that **READ_ACL** and **WRITE_ACL** always be granted to the object owner. Although this contradicts *NFS Version 4 Protocol*, it is viewed that this is an area where users would otherwise erroneously leave an ACL that only privileged users could change. Since ACLs are themselves file attributes, **READ_ATTR** and **WRITE_ATTR** are similarly granted to the owner. Since it would not make sense to then prevent the owner from accessing the ACL from a non-AIX node, GPFS has implemented this exception everywhere.
5. AIX does not support the use of special name values other than **owner@**, **group@**, and **everyone@**. Therefore, these are the only valid special name values for use in GPFS NFS V4 ACLs as well.
6. NFS V4 allows ACL entries that grant users (or groups) permission to change the owner or owning group of the file (for example, with the **chown** command). For security reasons, GPFS now restricts this so that non-privileged users may only **chown** such a file to themselves (becoming the owner) or to a group that they are a member of.
7. Windows does not support NFS V4 ACLs.
8. If a file system is to be exported over NFS V4/Linux, then it *must* be configured to support POSIX ACLs (with the **-k all** or **-k posix** option of the **mmcrfs** command). This is because NFS V4 Linux servers only handle ACLs properly if they are stored in GPFS as POSIX ACLs. For more information, see "Linux ACLs and extended attributes."
9. Concurrent Samba, AIX NFS servers, and GPFS Windows nodes in the cluster are allowed. NFS V4 ACLs can be stored in GPFS file systems using Samba exports, NFS V4 AIX servers, GPFS Windows nodes, **aclput**, and **mmputacl**. However, clients of Linux V4 servers will not be able to see these ACLs, just the permissions from the mode.

For more information about GPFS ACLs and NFS export, see Chapter 19, "Managing GPFS access control lists," on page 245.

## Linux ACLs and extended attributes

NFS V4 uses the existing POSIX ACLs and the extended attribute support in Linux that is supported by GPFS.

Although the NFS V4 protocol defines a richer ACL model similar to Windows ACLs, the Linux implementation maps those ACLs to POSIX ACLs before passing them to the underlying file system. This mapping is done in nfsd. This means that if you set an ACL from a client and then fetch it back, you will see that what the server returns is not exactly what you set. This is because the ACL you set was converted to a POSIX ACL and then back again.

NFS V4 ACLs are more fine-grained than POSIX ACLs, so the POSIX-to-NFS V4 translation is close to perfect, but the NFS V4-to-POSIX translation is not. The NFS V4 server attempts to err on the side of

mapping to a stricter ACL. There is a very small set of NFS V4 ACLs that the server rejects completely (for example, any ACL that attempts to explicitly DENY rights to read attributes), but otherwise, the server tries very hard to accept ACLs and map them as best it can.

ACLs that are set through AIX/NFS V4 and Windows nodes are written as NFS V4 ACLs to GPFS, whereas ACLs that are set through Linux/NFS V4 are written as POSIX ACLs to GPFS. Currently, GPFS does not provide an interface to convert on-disk NFS V4 ACLs to POSIX ACLs. This means that if ACLs are written through either AIX/NFS V4 or Windows, they cannot be read by Linux/NFS V4. In this case, a Linux NFS V4 server constructs an ACL from the permission mode bits only and ignores the ACL on the file.

## General NFS V4 Linux exceptions and limitations

GPFS has some NFS V4 Linux exceptions and limitations.

Specifically, Windows-based NFS V4 clients are not supported with Linux NFS V4 servers because of their use of share modes. In particular, GPFS does not support:

- A Windows NFS V4 client talking to multinode GPFS, where one of the nodes runs Linux. Since Windows open() allows share modes to be specified, these can be mapped directly to the NFS V4 share modes and passed to the server.
- A multinode GPFS cluster exported over NFS V4 and Samba, where one of the nodes runs Linux. Since Samba can enforce share modes throughout the cluster, interoperability will be broken with NFS V4, which cannot.
- An application that can be written to directly invoke the NFS V4 protocol. Since POSIX does not allow share modes to be specified, this is one way a UNIX (AIX or Linux) application can indicate share modes.
- Applications running over NFS V4 that map certain operations to share reservations. An AIX NFS V4 client maps the open(F_EXCL) call to an open(SHARE_DENY_READ) over NFS V4.

### IBM CES NFS stack limitations

NFSV4.0 limitations are described here:

- Changes to the IBM NFSv4.0 exports config and NFS global config are not dynamic. NFS services will automatically restart during the execution of the **mmnfs export change**, **mmnfs export load**, and **mmnfs config change** commands. During this time, an NFS client with a soft mount might lose connectivity. This may result in an application failure on the client node. An NFS client with a hard mount might "stall" during the NFS restart.
- Whenever NFS is restarted, a grace period will ensue. The default grace period is 90 seconds. If NFS export changes or NFS config changes are performed sequentially, then NFS services will be restarted multiple times, leading to a cumulative extended grace period. This might prevent NFS clients from reclaiming their locks, possibly leading to an application failure on the client node.
- To make multiple changes to existing NFS exports, do either of the following:
  - Wait for a lease period of more than 90 seconds before changing the next export.

    **Note:** The applications will see delayed NFS response due to NFS restarts and the associated grace periods.
  - Allow for a service window and modify the exports while manually restarting NFS services:
    1. **mmces service stop nfs -a**
    2. **mmnfs export change**
    3. **mmces service start nfs -a**

    **Note:** This approach requires a brief change window, and may be the preferred method when there are many changes planned for NFS exports.

| • Exporting symbolic links is not supported in CES NFS.

## Considerations for the use of direct I/O (O_DIRECT)

A file opened in the **O_DIRECT** mode ("direct I/O"), GPFS transfers data directly between the user buffer and the file on the disk.

Using direct I/O may provide some performance benefits in the following cases:

- The file is accessed at random locations.
- There is no access locality.

Direct transfer between the user buffer and the disk can only happen if all of the following conditions are true:

- The number of bytes transferred is a multiple of 512 bytes.
- The file offset is a multiple of 512 bytes.
- The user memory buffer address is aligned on a 512-byte boundary.

When these conditions are *not* all true, the operation will still proceed but will be treated more like other normal file I/O, with the **O_SYNC** flag that flushes the dirty buffer to disk.

When these conditions *are* all true, the GPFS pagepool is not used because the data is transferred directly; therefore, an environment in which most of the I/O volume is due to direct I/O (such as in databases) will not benefit from a large pagepool. Note, however, that the pagepool still needs to be configured with an adequate size, or left at its default value, because the pagepool is also used to store file metadata (especially for the indirect blocks required for large files).

With direct I/O, the application is responsible for coordinating access to the file, and neither the overhead nor the protection provided by GPFS locking mechanisms plays a role. In particular, if two threads or nodes perform direct I/O concurrently on overlapping portions of the file, the outcome is undefined. For example, when multiple writes are made to the same file offsets, it is undetermined which of the writes will be on the file when all I/O is completed. In addition, if the file has data replication, it is not guaranteed that all the data replicas will contain the data from the same writer. That is, the contents of each of the replicas could diverge.

Even when the I/O requests are aligned as previously listed, in the following cases GPFS will not transfer the data directly and will revert to the slower buffered behavior:

- The write causes the file to increase in size.
- The write is in a region of the file that has been preallocated (via **gpfs_prealloc()**) but has not yet been written.
- The write is in a region of the file where a "hole" is present; that is, the file is sparse and has some unallocated regions.

When direct I/O requests are aligned but none of the previously listed conditions (that would cause the buffered I/O path to be taken) are present, handling is optimized this way: the request is completely handled in kernel mode by the GPFS kernel module, without the GPFS daemon getting involved. Any of the following conditions, however, will still result in the request going through the daemon:

- The I/O operation needs to be served by an NSD server.
- The file system has data replication. In the case of a write operation, the GPFS daemon is involved to produce the log records that ensure that the replica contents are identical (in case of a failure while writing the replicas to disk).
- The operation is performed on the Windows operating system.

Note that setting the **O_DIRECT** flag on an open file with **fcntl (fd, F_SETFL,** [..]**)**, which may be allowed on Linux, is ignored in a GPFS file system.

## NFS protocol node limitations

NFS protocol nodes have conflicting NLM locking services between the Linux kernel and CES NFS.

When mounting an NFSv3 file system on a protocol node, the Linux kernel lockd daemon registers with the rpcbind, preventing the CES NFS lock service from taking effect. If you need to mount an NFSv3 file system on a CES NFS protocol node, use the **-o nolock mount** option to prevent invoking Linux kernel lockd daemon.

# Chapter 21. Accessing a remote GPFS file system

GPFS allows users shared access to files in either the cluster where the file system was created, or other GPFS clusters. File system access by the cluster where the file system was created is implicit.

The ability to access and mount GPFS file systems owned by other clusters in a network of sufficient bandwidth is accomplished using the **mmauth**, **mmremotecluster** and **mmremotefs** commands. Each site in the network is managed as a separate cluster, while allowing shared file system access.

The cluster owning the file system is responsible for administering the file system and granting access to other clusters on a per cluster basis. After access to a particular file system has been granted to nodes in another GPFS cluster, the nodes can mount the file system and perform data operations as if the file system were locally owned.

Each node in the GPFS cluster requiring access to another cluster's file system must be able to open a TCP/IP connection to every node in the other cluster.

Nodes in two separate remote clusters mounting the same file system are not required to be able to open a TCP/IP connection to each other. For example, if a node in `clusterA` mounts a file system from `clusterB`, and a node in `clusterC` desires to mount the same file system, nodes in `clusterA` and `clusterC` do not have to communicate with each other.

Each node in the GPFS cluster requiring file system access must have one of the following:

- A virtual connection to the file system data through an NSD server (refer to Figure 6 on page 282).
- A physical connection to the disks containing file system data (refer to Figure 7 on page 282).

In this example, network connectivity is required from the nodes in `clusterB` to all the nodes in `clusterA` even if the nodes in `clusterB` can access the disks in `clusterA` directly.

**Note:** Even when remote nodes have direct connectivity to the SAN, they will still use a connection through an NSD server for any NSDs that have been configured with Persistent Reserve (PR). If you want the remote nodes to access the disks through their direct connection to the SAN, you must ensure that PR is not enabled on the NSDs. See “Enabling and disabling Persistent Reserve” on page 122.

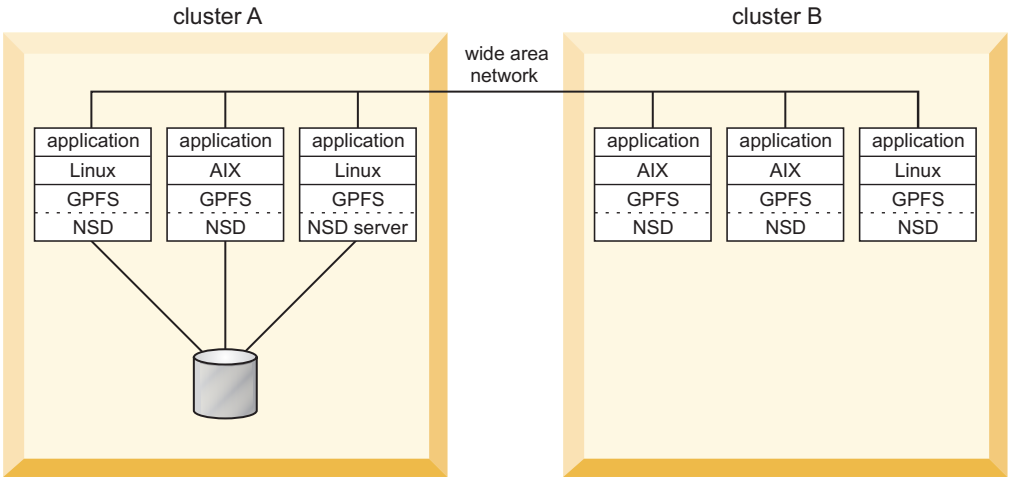



*Figure 6. Remote mount of a file system using NSD server access*

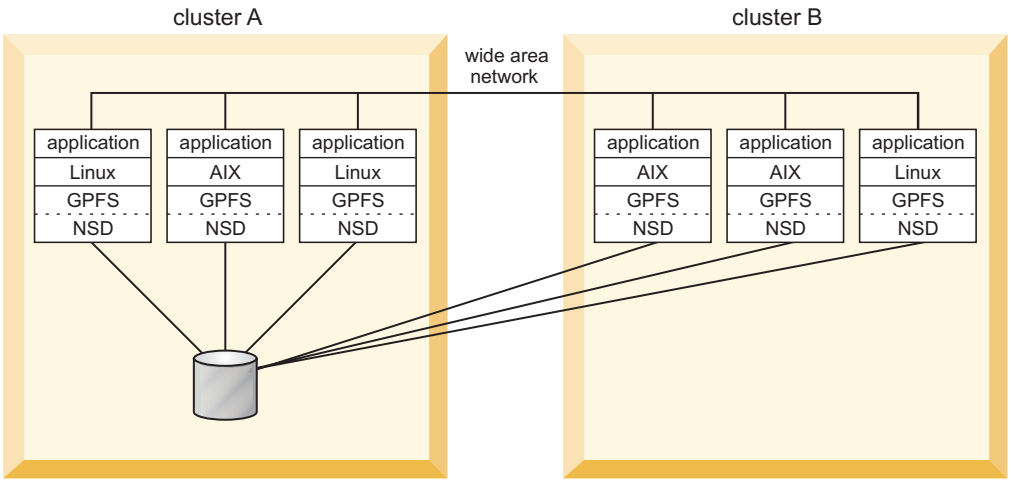



*Figure 7. Remote mount of a file system using SAN-attached disks*

Figure 8 on page 283 illustrates a multi-cluster configuration with multiple NSD servers. In this configuration:

- The two nodes in Cluster 1 are defined as the NSD servers (you can have up to eight NSD server nodes).
- All three clusters are connected with Gigabit Ethernet.
- Cluster 1 shares an InfiniBand switch network with Cluster 2 and an InfiniBand switch network with Cluster 3.

**Note:** Protocol support is not available in a multi-cluster configuration.

In order to take advantage of the fast networks and to use the nodes in Cluster 1 as NSD servers for Cluster 2 and Cluster 3, you must configure a subnet for each of the supported clusters. For example issuing the command:

- **mmchconfig subnets="<***IB_Network_1***> <***IB_Network_1***>/Cluster1"** in Cluster 2 allows nodes $N_2$ through $N_x$ to use $N_1$ as an NSD server with InfiniBand Network 1 providing the path to the data.
- **mmchconfig subnets="<***IB_Network_2***> <***IB_Network_2***>/Cluster1"** in Cluster 3 allows nodes $N_{2+x}$ through $N_{y+x}$ to use $N_{1+x}$ as an NSD server with InfiniBand Network 2 providing the path to the data.

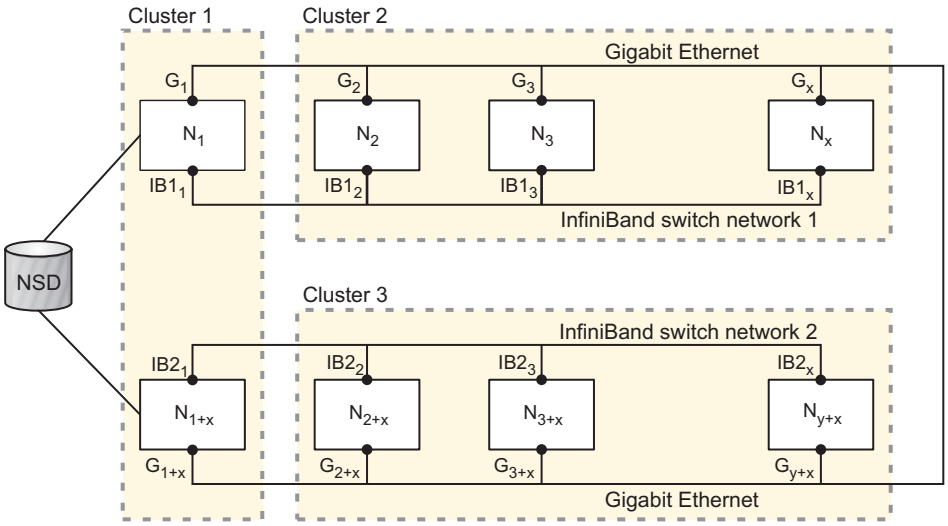



*Figure 8. Multi-cluster configuration with multiple NSD servers*

When you implement file access from other clusters, consider these topics:

- "Remote user access to a GPFS file system"
- "Mounting a remote GPFS file system" on page 284
- "Managing remote access to a GPFS file system" on page 286
- "Using remote access with public and private IP addresses" on page 287
- "Using multiple security levels for remote access" on page 289
- "Changing security keys with remote access" on page 290
- "Important information about remote access" on page 292

## Remote user access to a GPFS file system

In a cluster environment that has a single user identity namespace, all nodes have user accounts set up in a uniform manner. This is usually accomplished by having equivalent **/etc/passwd** and **/etc/group** files on all nodes in the cluster.

For consistency of ownership and access control, a uniform user identity namespace is preferred. For example, if user Jane Doe has an account on nodeA with the user name **janedoe** and user ID **1001** and group ID **500**, on all other nodes in the same cluster Jane Doe will have an account with the same user and group IDs. GPFS relies on this behavior to perform file ownership and access control tasks.

If a GPFS file system is being accessed from a node belonging to another GPFS cluster, the assumption about the uniform user account infrastructure might no longer be valid. Since different clusters can be administered by different organizations, it is possible for each of the clusters to have a unique set of user accounts. This presents the problem of how to permit users to access files in a file system owned and served by another GPFS cluster. In order to have such access, the user must be somehow known to the other cluster. This is usually accomplished by creating a user account in the other cluster, and giving this account the same set of user and group IDs that the account has in the cluster where the file system was created.

To continue with this example, Jane Doe would need an account with user ID **1001** and group ID **500** created in every other GPFS cluster from which remote GPFS file system access is desired. This approach is commonly used for access control in other network file systems, (for example, NFS or AFS™), but might pose problems in some situations.

For example, a problem arises if Jane Doe already has an account in some other cluster, but the user ID associated with this account is not **1001**, and another user in the other cluster has user ID **1001**. It would require a considerable effort on the part of system administrator to ensure that Jane Doe's account has the same set of IDs on all clusters. It is more desirable to be able to use the existing accounts without having to make changes. GPFS helps to solve this problem by optionally performing user ID and group ID remapping internally, using user-supplied helper applications. For a detailed description of the GPFS user ID remapping convention, 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).

Access from a remote cluster by a root user presents a special case. It is often desirable to disallow root access from a remote cluster while allowing regular user access. Such a restriction is commonly known as root squash. A root squash option is available when making a file system available for mounting by other clusters using the **mmauth** command. This option is similar to the NFS root squash option. When enabled, it causes GPFS to squash superuser authority on accesses to the affected file system on nodes in remote clusters.

This is accomplished by remapping the credentials: user id (UID) and group id (GID) of the root user, to a UID and GID specified by the system administrator on the home cluster, for example, the UID and GID of the user nobody. In effect, root squashing makes the root user on remote nodes access the file system as a non-privileged user.

Although enabling root squash is similar to setting up UID remapping, there are two important differences:

1. While enabling UID remapping on remote nodes is an option available to the remote system administrator, root squashing need only be enabled on the local cluster, and it will be enforced on remote nodes. Regular UID remapping is a user convenience feature, while root squashing is a security feature.
2. While UID remapping requires having an external infrastructure for mapping between local names and globally unique names, no such infrastructure is necessary for enabling root squashing.

When both UID remapping and root squashing are enabled, root squashing overrides the normal UID remapping mechanism for the root user.

## Mounting a remote GPFS file system

This is an example of how to mount a file system owned and served by another IBM Spectrum Scale cluster. The `gpfs.gskit` package must be installed on all nodes in the involved clusters before using these instructions.

The procedure to set up remote file system access involves the generation and exchange of authorization keys between the two clusters. In addition, the administrator of the GPFS cluster that owns the file system needs to authorize the remote clusters that are to access it, while the administrator of the GPFS cluster that seeks access to a remote file system needs to define to GPFS the remote cluster and file system whose access is desired.

| **Note:** For more information on CES cluster setup, see "CES cluster setup" on page 393.

In this example, **cluster1** is the name of the cluster that owns and serves the file system to be mounted, and **cluster2** is the name of the cluster that desires to access the file system.

**Note:** The following example uses AUTHONLY as the authorization setting. When you specify AUTHONLY for authentication, GPFS checks network connection authorization. However, data sent over the connection is not protected.

1. On **cluster1**, the system administrator issues the **mmauth** command to generate a public/private key pair. The key pair is placed in **/var/mmfs/ssl**:

   ```
   mmauth genkey new
   ```

2. On **cluster1**, the system administrator enables authorization by issuing:

   ```
   mmauth update . -l AUTHONLY
   ```

3. The system administrator of **cluster1** now gives the file **/var/mmfs/ssl/id_rsa.pub** to the system administrator of **cluster2**, who desires to access the **cluster1** file systems. This operation requires the two administrators to coordinate their activities, and must occur outside of the GPFS command environment.
4. On **cluster2**, the system administrator issues the **mmauth** command to generate a public/private key pair. The key pair is placed in **/var/mmfs/ssl**:

   ```
   mmauth genkey new
   ```

5. On **cluster2**, the system administrator enables authorization by issuing:

   ```
   mmauth update . -l AUTHONLY
   ```

6. The system administrator of **cluster2** gives file **/var/mmfs/ssl/id_rsa.pub** to the system administrator of **cluster1**. This operation requires the two administrators to coordinate their activities, and must occur outside of the GPFS command environment.
7. On **cluster1**, the system administrator issues the **mmauth add** command to authorize **cluster2** to mount file systems owned by **cluster1** utilizing the key file received from the administrator of **cluster2**:

   ```
   mmauth add cluster2 -k cluster2_id_rsa.pub
   ```

   where:

   *cluster2*
   Is the real name of **cluster2** as given by the **mmlscluster** command on a node in **cluster2**.

   *cluster2_id_rsa.pub*
   Is the name of the file obtained from the administrator of **cluster2** in Step 6.
8. On **cluster1**, the system administrator issues the **mmauth grant** command to authorize **cluster2** to mount specific file systems owned by **cluster1**:

   ```
   mmauth grant cluster2 -f /dev/gpfs
   ```

9. On **cluster2**, the system administrator now must define the cluster name, contact nodes and public key for **cluster1**:

   ```
   mmremotecluster add cluster1 -n node1,node2,node3 -k cluster1_id_rsa.pub
   ```

   where:

   *cluster1*
   Is the real name of **cluster1** as given by the **mmlscluster** command on a node in **cluster1**.

   *node1*, *node2*, **and** *node3*
   Are nodes in **cluster1**. The hostname or IP address that you specify must refer to the communications adapter that is used by GPFS as given by the **mmlscluster** command on a node in **cluster1**.

   *cluster1_id_rsa.pub*
   Is the name of the file obtained from the administrator of **cluster1** in Step 3.

   This permits the cluster desiring to mount the file system a means to locate the serving cluster and ultimately mount its file systems.
10. On **cluster2**, the system administrator issues one or more **mmremotefs** commands to identify the file systems in **cluster1** that are to be accessed by nodes in **cluster2**:

    ```
    mmremotefs add /dev/mygpfs -f /dev/gpfs -C cluster1 -T /mygpfs
    ```

    where:

*/dev/mygpfs*
Is the device name under which the file system will be known in **cluster2**.

*/dev/gpfs*
Is the actual device name for the file system in **cluster1**.

*cluster1*
Is the real name of **cluster1** as given by the **mmlscluster** command on a node in **cluster1**.

*/mygpfs*
Is the local mount point in **cluster2**.

11. On **cluster2**, the command:

```
mmmount /dev/mygpfs
```

then mounts the file system.

Table 34 summarizes the commands that the administrators of the two clusters need to issue so that the nodes in **cluster2** can mount the remote file system **fs1**, owned by **cluster1**, assigning **rfs1** as the local name with a mount point of **/rfs1**.

*Table 34. Summary of commands to set up cross-cluster file system access.*

| **cluster1** | **cluster2** |
|---|---|
| **mmauth genkey new** | **mmauth genkey new** |
| **mmauth update . -l AUTHONLY** | **mmauth update . -l AUTHONLY** |
| Exchange public keys (file **/var/mmfs/ssl/id_rsa.pub**) | |
| **mmauth add** *cluster2 ...* | **mmremotecluster add** *cluster1 ...* |
| **mmauth grant** *cluster2* **-f** fs1 ... | **mmremotefs add** rfs1 **-f** fs1 **-C** cluster1 **-T** /rfs1 |

## Managing remote access to a GPFS file system

This is an example of how to manage remote access to GPFS file systems.

To see a list of all clusters authorized to mount file systems owned by **cluster1**, the administrator of **cluster1** issues this command:

```
mmauth show
```

To authorize a third cluster, say **cluster3**, to access file systems owned by **cluster1**, the administrator of **cluster1** issues this command:

```
mmauth add cluster3 -k cluster3_id_rsa.pub
mmauth grant cluster3 -f /dev/gpfs1
```

To subsequently revoke **cluster3** authorization to access a specific file system **gpfs1** owned by **cluster1**, the administrator of **cluster1** issues this command:

```
mmauth deny cluster3 -f /dev/gpfs1
```

To completely revoke **cluster3** authorization to access file systems owned by **cluster1**, the administrator of **cluster1** issues this command:

```
mmauth delete cluster3
```

## Using remote access with public and private IP addresses

GPFS permits the use of both public and private IP address. Private IP addresses are typically used to communicate on private networks.

Private IP addresses are on one of these subnets:

- 10.0.0.0
- 172.16.0.0
- 192.168.0.0

See RFC 1597 - Address Allocation for Private Internets (www.ip-doc.com/rfc/rfc1597) for more information.

Use the **mmchconfig** command, **subnets** attribute, to specify the private IP addresses to be accessed by GPFS.

Figure 9 on page 289 describes an AIX cluster named **CL1** with nodes named **CL1N1**, **CL1N2**, and so forth, a Linux cluster named **CL2** with nodes named **CL2N1**, **CL2N2**, and another Linux cluster named **CL3** with a node named **CL3N1**. Both Linux clusters have public Ethernet connectivity, and a Gigabit Ethernet configured with private IP addresses (10.200.0.1 through 10.200.0.24), not connected to the public Ethernet. The InfiniBand Switch on the AIX cluster **CL1** is configured using public IP addresses on the 7.2.24/13 subnet and is accessible from the outside.

With the use of both public and private IP addresses for some of the nodes, the setup works as follows:

1. All clusters must be created using host names or IP addresses that correspond to the public network.
2. Using the **mmchconfig** command for the **CL1** cluster, add the attribute: **subnets=7.2.24.0**.

   This allows all **CL1** nodes to communicate using the InfiniBand Switch. Remote mounts between **CL2** and **CL1** will use the public Ethernet for TCP/IP communication, since the **CL2** nodes are not on the 7.2.24.0 subnet.

   GPFS assumes subnet specifications for private networks are independent between clusters (private networks are assumed not physically connected between clusters). The remaining steps show how to indicate that a private network is shared between clusters.
3. Using the **mmchconfig** command for the **CL2** cluster, add the **subnets='10.200.0.0/CL2.kgn.ibm.com;CL3.kgn.ibm.com'** attribute. Alternatively, regular expressions are allowed here, such as **subnets='10.200.0.0/CL[23].kgn.ibm.com'**. See note 2 on page 288 for the syntax allowed for the regular expressions.

   This attribute indicates that the private 10.200.0.0 network extends to all nodes in clusters **CL2** or **CL3**. This way, any two nodes in the **CL2** and **CL3** clusters can communicate through the Gigabit Ethernet.

   This setting allows all **CL2** nodes to communicate over their Gigabit Ethernet. Matching **CL3.kgn.ibm.com** with the cluster list for 10.200.0.0 allows remote mounts between clusters **CL2** and **CL3** to communicate over their Gigabit Ethernet.
4. Using the **mmchconfig** command for the **CL3** cluster, add the **subnets='10.200.0.0/CL3.kgn.ibm.com;CL2.kgn.ibm.com'** attribute, alternatively **subnets='10.200.0.0/CL[32].kgn.ibm.com'**.

   This attribute indicates that the private 10.200.0.0 network extends to all nodes in clusters **CL2** or **CL3**. This way, any two nodes in the **CL2** and **CL3** clusters can communicate through the Gigabit Ethernet.

   Matching of **CL3.kgn.ibm.com** with the cluster list for 10.200.0.0 allows all **CL3** nodes to communicate over their Gigabit Ethernet, and matching **CL2.kgn.ibm.com** with that list allows remote mounts between clusters **CL3** and **CL2** to communicate over their Gigabit Ethernet.

Use the **subnets** attribute of the **mmchconfig** command when you wish the GPFS cluster to leverage additional, higher performance network connections that are available to the nodes in the cluster, or between clusters.

**Notes:**

1. Use of the **subnets** attribute does not ensure a highly available system. If the GPFS daemon is using the IP address specified by the **subnets** attribute, and that interface goes down, GPFS does not switch to the other network. You can use **mmdiag --network** to verify that the subnet is in fact being used.
2. Each subnet can be listed at most once in each cluster. For example, specifying:

   ```
   subnets='10.200.0.0/CL2.kgn.ibm.com 10.200.0.0/CL3.kgn.ibm.com'
   ```

   where the 10.200.0.0 subnet is listed twice, is not allowed. Therefore, subnets that span multiple clusters have to be assigned a cluster name pattern or a semicolon-separated cluster name list. It is possible to combine these, for example, items in semicolon-separated cluster lists can be plain names or regular expressions, as in the following:

   ```
   subnets='1.0.0.1/CL[23].kgn.ibm.com;OC.xyz.ibm.com'
   ```

   The following shows examples of patterns that are accepted:

   ```
   [af3] matches letters 'a' and 'f', and number 3
   [0-7] matches numbers 0, 1, ... 7
   [a-p0-7] matches letter a, b, ... p and numbers from 0 to 7 inclusive
   * matches any sequence of characters
   ? matches any (one) character
   ```

   If the **subnets** attribute lists multiple subnets, and there are multiple subnets in common between the local cluster and a given remote cluster, then the first subnet in common in the list is used for communications between the local and remote clusters. As an example, suppose that the **subnets** attribute is set as follows, on cluster **CL2.kgn.ibm.com**:

   ```
   subnets='10.200.0.0/CL[23].kgn.ibm.com 10.201.0.0/CL[23].kgn.ibm.com'
   ```

   If node **CL2N1** on cluster **CL2.kgn.ibm.com** has network interfaces with IP addresses 10.200.0.1 and 10.201.0.1, and node **CLN31** on cluster **CL3.kgn.ibm.com** has network interfaces with IP addresses 10.200.0.5 and 10.201.0.5, then the communication between these two nodes will flow over the 10.200.0.0 subnet, with **CL2N1** using the interface with IP address 10.200.0.1, and **CLN31** using the interface with IP address 10.200.0.5.

   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.

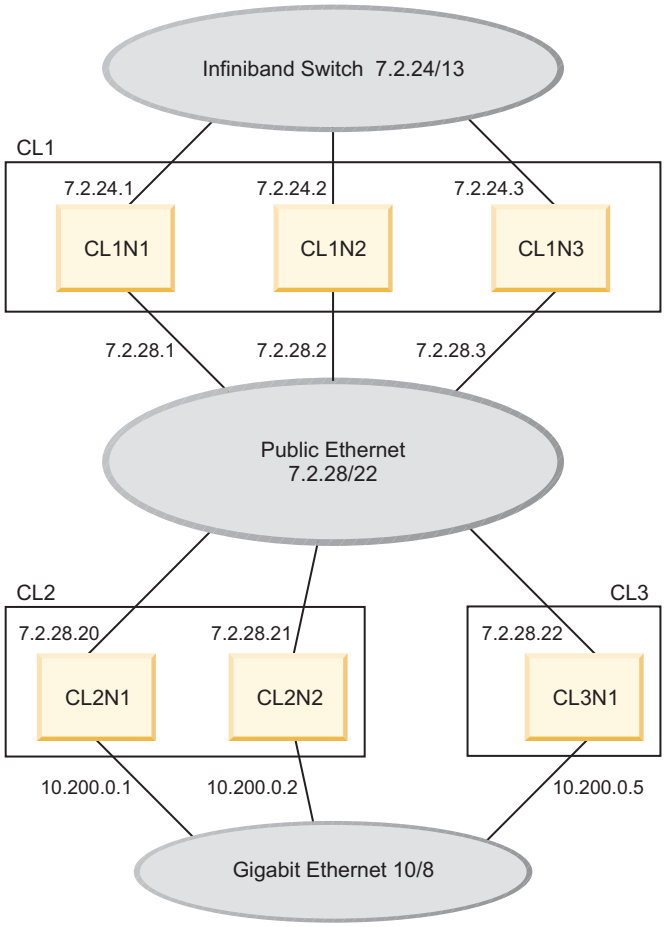



*Figure 9. Use of public and private IP addresses in three GPFS clusters*

## Using multiple security levels for remote access

A cluster that owns a file system whose access is to be permitted from other clusters, can designate a different security level for each connecting cluster.

When multiple security levels are specified, the following rule applies: each connection uses the security level of the connecting node, unless that security level is **AUTHONLY**. In this case, the security level of the node accepting the connection is used instead. This means that a connection will use **AUTHONLY** if and only if both nodes exist in clusters that are required to use security method **AUTHONLY**.

To specify a different security level for different clusters requesting access to a given cluster, use the **mmauth -l** *cipherList* command. Several examples follow to illustrate:

1. In this example, **cluster1** and **cluster2** are located on the same trusted network, and **cluster3** is connected to both of them with an untrusted network. The system administrator chooses these security levels:
   - A *cipherList* of **AUTHONLY** for connections between **cluster1** and **cluster2**
   - A *cipherList* of **AES128-SHA** for connections between **cluster1** and **cluster3**
   - A *cipherList* of **AES128-SHA** for connections between **cluster2** and **cluster3**

   The administrator of **cluster1** issues these commands:

   ```
   mmauth add cluster2 -k keyFile -l AUTHONLY
   mmauth add cluster3 -k keyFile -l AES128-SHA
   ```

2. In this example, **cluster2** is accessing file systems owned by **cluster1** using a *cipherList* of **AUTHONLY**, but the administrator of **cluster1** has decided to require a more secure *cipherList*. The administrator of **cluster1** issues this command:

   ```
   mmauth update cluster2 -l AES128-SHA
   ```

   Existing connections will be upgraded from **AUTHONLY** to **AES128-SHA**.

## Changing security keys with remote access

When working with GPFS file systems accessed by other GPFS clusters, it might be necessary to generate a new public/private access key. This can be done without disturbing existing connections, provided the following procedure is followed.

To accomplish this, the cluster that owns and serves the file system is made to temporarily have two access keys (referred to as the 'old key' and the 'new key'), which are both valid at the same time. The clusters currently accessing the file system can then change from the old key to the new key without interruption of file system access.

In this example, **cluster1** is the name of the cluster that owns and serves a file system, and **cluster2** is the name of the cluster that has already obtained access to this file system, and is currently using it. Here, the system administrator of **cluster1** changes the access key without severing the connection obtained by **cluster2**.

1. On **cluster1**, the system administrator issues the **mmauth genkey new** command to generate a new public/private access key pair. The key pair is placed in **/var/mmfs/ssl**:

   ```
   mmauth genkey new
   ```

   After this command is issued, **cluster1** will have two keys (referred to as the 'old key' and the 'new key') that can both be used to access **cluster1** file systems.
2. The system administrator of **cluster1** now gives the file **/var/mmfs/ssl/id_rsa.pub** (that contains the new key) to the system administrator of **cluster2**, who desires to continue to access the **cluster1** file systems. This operation requires the two administrators to coordinate their activities, and must occur outside of the GPFS command environment.
3. On **cluster2**, the system administrator issues the **mmremotecluster update** command to make the new key known to his system:

   ```
   mmremotecluster update cluster1 -k cluster1_id_rsa.pub
   ```

   where:

   *cluster1*
   : Is the real name of **cluster1** as given by the **mmlscluster** command on a node in **cluster1**.

   *cluster1_id_rsa.pub*
   : Is the name of the file obtained from the administrator of **cluster1** in Step 2.

   This permits the cluster desiring to mount the file system to continue mounting file systems owned by **cluster1**.
4. On **cluster1**, the system administrator verifies that all clusters desiring to access **cluster1** file systems have received the new key and activated it using the **mmremotecluster update** command.
5. On **cluster1**, the system administrator issues the **mmauth genkey commit** command to commit the new key as the only valid access key. The old key will no longer be accepted once this command completes successfully:

   ```
   mmauth genkey commit
   ```

   Once the new public key has been committed, the old public key will no longer be accepted. As a result, any remote cluster administrator who has not been given the new key (see the preceding Step 2) and run **mmremotecluster update** (see the preceding Step 3) will no longer be able to mount file systems owned by **cluster1**.

Similarly, the administrator of **cluster2** might decide to change the access key for **cluster2**:

1. On **cluster2**, the system administrator issues the **mmauth genkey new** command to generate a new public/private access key pair. The key pair is placed in **/var/mmfs/ssl**:

   ```
   mmauth genkey new
   ```

   After this command is issued, **cluster2** will have two keys (referred to as the 'old key' and the 'new key') that can both be used when a connection is established to any of the nodes in **cluster2**.
2. The system administrator of **cluster2** now gives the file **/var/mmfs/ssl/id_rsa.pub** (that contains the new key) to the system administrator of **cluster1**, the owner of the file systems. This operation requires the two administrators to coordinate their activities, and must occur outside of the GPFS command environment.
3. On **cluster1**, the system administrator issues the **mmauth update** command to make the new key known to his system:

   ```
   mmauth update cluster2 -k cluster2_id_rsa.pub
   ```

   where:

   *cluster2*
   Is the real name of **cluster2** as given by the **mmlscluster** command on a node in **cluster2**.

   *cluster2_id_rsa.pub*
   Is the name of the file obtained from the administrator of **cluster2** in Step 2.

   This permits the cluster desiring to mount the file system to continue mounting file systems owned by **cluster1**.
4. The system administrator of **cluster2** verifies that the administrator of **cluster1** has received the new key and activated it using the **mmauth update** command.
5. On **cluster2**, the system administrator issues the **mmauth genkey commit** command to commit the new key as the only valid access key. The old key will no longer be accepted once this command completes successfully:

   ```
   mmauth genkey commit
   ```

# NIST compliance

The **nistCompliance** configuration variable allows the system administrator to restrict the set of available algorithms and key lengths to a subset of those approved by NIST.

## About this task

The **nistCompliance** variable applies to security transport (tscomm security, key retrieval) only, not to encryption, which always uses NIST-compliant mechanisms.

For the valid values for **nistCompliance**, see *mmchconfig command* in the *IBM Spectrum Scale: Command and Programming Reference* guide.

The **nistCompliance** configuration variable has been introduced on version 4.1. Clusters created prior to that release operate with the equivalent of that variable being set to **off**. Similarly, clusters created on prior versions and which are migrated to 4.1 will have **nistCompliance** set to **off**.

## Remote Mounts and version 3.5 clusters

A cluster created on version 4.1 or higher, and operating with **nistCompliance** set to **SP800-131A**, will be unable to remote-mount a file system from a version 3.5 cluster, since the 4.1 cluster will not accept the key from the latter, which is not NIST SP800-131A-compliant. To allow the version 4.1 cluster to remote-mount the version 3.5 cluster, issue the

```
mmchconfig nistCompliance=off
```

command on the version 4.1 cluster, before the **mmremotecluster add** command can be issued. The key exchange will work even if the version 4.1 cluster already has a NIST-compliant key.

### Updating a cluster to nistCompliance SP800-131A

A cluster upgraded from prior versions may have the **nistCompliance** set to **off** and may be operating with keys which are not NIST SP800-131A-compliant. To upgrade the cluster to operate in NIST SP800-131A mode, the following procedure should be followed:

From a node in the cluster which is running version 4.1 or later, issue:

```
mmauth genkey new
mmauth genkey commit
```

If remote clusters are present, follow the procedure described in the "Changing security keys with remote access" on page 290 section (under Chapter 21, "Accessing a remote GPFS file system," on page 281) to update the key on the remote clusters.

Once all nodes in the cluster are running at least version 4.1, run the following command from one of the nodes in the cluster:

```
mmchconfig release=LATEST
```

From one of the nodes in the cluster, run the following command:

```
mmchconfig nistCompliance=SP800-131A
```

## Important information about remote access

There is some additional information about this topic that you should take into consideration.

When working with GPFS file systems accessed by nodes that belong to other GPFS clusters, consider the following points:

1. A file system is administered only by the cluster where the file system was created. Other clusters may be allowed to mount the file system, but their administrators cannot add or delete disks, change characteristics of the file system, enable or disable quotas, run the **mmfsck** command, and so forth. The only commands that other clusters can issue are list type commands, such as: **mmlsfs**, **mmlsdisk**, **mmlsmount**, and **mmdf**.
2. 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 the administrator of **cluster1** (the owner of file system **gpfs1**) decides to delete it or rename it, the information for **gpfs1** in **cluster2** becomes obsolete, and an attempt to mount **gpfs1** from **cluster2** will fail. It is assumed that when such changes take place, the two administrators will inform each other. The administrator of **cluster2** can then use the **update** or **delete** options of the **mmremotefs** command to make the appropriate changes.
3. If the names of the contact nodes change, the name of the cluster changes, or the public key file changes, use the **update** option of the **mmremotecluster** command to reflect the changes.
4. Use the **show** option of the **mmremotecluster** and **mmremotefs** commands to display the current information about remote clusters and file systems.
5. If the cluster that owns a file system has a **maxblocksize** configuration parameter that is different from the **maxblocksize** configuration parameter of the cluster that desires to mount a file system, a mismatch may occur and file system mount requests may fail with messages to this effect. Check your **maxblocksize** configuration parameters on both clusters using the **mmlsconfig** command. Correct any discrepancies with the **mmchconfig** command.

# Chapter 22. Information lifecycle management for IBM Spectrum Scale

IBM Spectrum Scale can help you achieve information lifecycle management (ILM) efficiencies through powerful policy-driven automated tiered storage management. With the ILM toolkit, you can manage sets of files and pools of storage, and you can automate the management of file data.

Using these tools, GPFS can automatically determine where to physically store your data regardless of its placement in the logical directory structure. Storage pools, filesets and user-defined policies provide the ability to match the cost of your storage resources to the value of your data.

**Note:** This feature is available with IBM Spectrum Scale Standard Edition or higher.

GPFS policy-based ILM tools allow you to:

- Create *storage pools* to provide a way to partition a file system's storage into collections of disks or a redundant array of independent disks (RAIDs) with similar properties that are managed together as a group. GPFS has three types of storage pools:
  - A required **system** storage pool that you create and manage through GPFS
  - Optional user storage pools that you create and manage through GPFS
  - Optional external storage pools that you define with GPFS policy rules and manage through an external application such as IBM Spectrum Protect
- Create *filesets* to provide a way to partition the file system namespace to allow administrative operations at a finer granularity than that of the entire file system. See "Filesets" on page 336.
- Create *policy rules* based on data attributes to determine initial file data placement and manage file data placement throughout the life of the file. See "Policies for automating file management" on page 300.

To work with ILM in the GUI, click **Files** > **Information Lifecycle**.

Use the following information to create and manage information lifecycle management policies in IBM Spectrum Scale:

## Storage pools

Physically, a *storage pool* is a collection of disks or RAID arrays. Storage pools also allow you to group multiple storage systems within a file system.

Using storage pools, you can create tiers of storage by grouping storage devices based on performance, locality, or reliability characteristics. For example, one pool could be an enterprise class storage system that hosts high-performance Fibre Channel disks and another pool might consist of numerous disk controllers that host a large set of economical SATA disks.

There are two types of storage pools in GPFS, internal storage pools and external storage pools. Internal storage pools are managed within GPFS. External storage pools are managed by an external application such as IBM Spectrum Protect. For external storage pools, GPFS provides tools that allow you to define an interface that your external storage manager uses to access your data. GPFS does not manage the data placed in external storage pools. Instead, GPFS manages the movement of data to and from external storage pools. Storage pools allow you to perform complex operations such as moving, mirroring, or deleting files across multiple storage devices, providing storage virtualization and a single management context.

Internal GPFS storage pools are meant for managing online storage resources. External storage pools are intended for use as near-line storage and for archival and backup operations. However, both types of storage pools provide you with a method to partition file system storage for considerations such as:

- Improved price-performance by matching the cost of storage to the value of the data
- Improved performance by:
  - Reducing the contention for premium storage
  - Reducing the impact of slower devices
  - Allowing you to retrieve archived data when needed
- Improved reliability by providing for:
  - Replication based on need
  - Better failure containment
  - Creation of new storage pools as needed

For additional information, refer to:

- "Internal storage pools"
- "External storage pools" on page 299

## Internal storage pools

Internal GPFS storage pools are controlled by GPFS policies and commands. There are two types of internal GPFS storage pools, the required system storage pool and up to seven optional user storage pools. The system storage pool contains metadata for each file and may also contain user data. User storage pools can only contain user data.

The internal GPFS storage pool to which a disk belongs is specified as an attribute of the disk in the GPFS cluster. You specify the disk attributes as a field in each disk descriptor when you create the file system or when adding disks to an existing file system. GPFS allows a maximum of eight internal storage pools per file system. One of these storage pools is the required **system** storage pool. The other seven internal storage pools are optional user storage pools.

GPFS assigns file data to internal storage pools under these circumstances:

- When the file is initially created; the storage pool is determined by the file placement policy that is in effect when at the time of file creation.
- When the attributes of the file, such as file size or access time, match the rules of a policy that directs GPFS to migrate the data to a different storage pool.

For additional information, refer to:

- "The system storage pool"
- "The system.log storage pool" on page 295
- "User storage pools" on page 295
- "Managing storage pools" on page 296

### The system storage pool

The **system** storage pool contains file system control structures, reserved files, directories, symbolic links, special devices, as well as the metadata associated with regular files, including indirect blocks, extended attributes, and so forth.

The **system** storage pool can also contain user data. There is only one **system** storage pool per file system, and it is automatically created when the file system is created.

**Important:** It is recommended that you use highly-reliable disks and replication for the **system** storage pool because it contains system metadata.

The amount of metadata grows as you add files to the system. Therefore, it is recommended that you monitor the **system** storage pool to ensure that there is always enough space to accommodate growth. The **system** storage pool typically requires a small percentage of the total storage capacity that GPFS manages. However, the percentage required by the **system** storage pool varies depending on your environment. You can monitor the amount of space available in the **system** storage pool with the **mmdf** command. If the available space in the system storage pool begins to run low, you can increase the available space by purging files or adding disks to the system storage pool.

## The system.log storage pool

By default the file system recovery log is stored in the system storage pool with file system metadata. The file system recovery log can also be placed in a dedicated pool that is called the **system.log** pool.

This storage pool must be created explicitly. It is highly recommended to only use storage that is as fast or even faster than what is used for the system storage pool. This recommendation is because of the high number of small synchronous data updates made to the recovery log. The block size for the **system.log** pool must be the same as the block size of the system pool.

The file system recovery log will only be stored in one pool.

## User storage pools

All user data for a file is stored in the assigned storage pool as determined by your file placement rules.

In addition, file data can be migrated to a different storage pool according to your file management policies. For more information on policies, see "Policies for automating file management" on page 300.

A user storage pool *only* contains the blocks of data (user data, for example) that make up a user file. GPFS stores the data that describes the files, called *file metadata*, separately from the actual file data in the **system** storage pool. You can create one or more user storage pools, and then create policy rules to indicate where the data blocks for a file should be stored.

### Tracking file access temperature within a storage pool

A file's access temperature is an attribute for policy that provides a means of optimizing tiered storage. File temperatures are a relative attribute, indicating whether a file is "hotter" or "colder" than the others in its pool. The policy can be used to migrate hotter files to higher tiers and colder files to lower. The access temperature is an exponential moving average of the accesses to the file. As files are accessed the temperature increases; likewise when the access stops the file cools. File temperature is intended to optimize nonvolatile storage, not memory usage; therefore, cache hits are not counted. In a similar manner, only user accesses are counted.

The access counts to a file are tracked as an exponential moving average. An unaccessed file loses a percentage of its accesses each period. The loss percentage and period are set via the configuration variables **fileHeatLossPercent** and **fileHeatPeriodMinutes**. By default, the file access temperature is not tracked. To use access temperature in policy, the tracking must first be enabled. To do this, set the two configuration variables as follows:

**fileHeatLossPercent**
: The percentage (between 0 and 100) of file access temperature dissipated over the **fileHeatPeriodMinutes** time. The default value is 10.

**fileHeatPeriodMinutes**
: The number of minutes defined for the recalculation of file access temperature. To turn on tracking, **fileHeatPeriodMinutes** must be set to a nonzero value. The default value is 0.

The following example sets **fileHeatPeriodMinutes** to 1440 (24 hours) and **fileHeatLossPercent** to 10, meaning that unaccessed files will lose 10% of their heat value every 24 hours, or approximately 0.4% every hour (because the loss is continuous and "compounded" geometrically):

```
mmchconfig fileheatperiodminutes=1440,fileheatlosspercent=10
```

**Note:** If the updating of the file access time (atime) is suppressed or if relative atime semantics are in effect, proper calculation of the file access temperature may be adversely affected.

File access temperature is tracked on a per-cluster basis, not on a per–file system basis.

Use **WEIGHT(FILE_HEAT)** with a policy **MIGRATE** rule to prioritize migration by file temperature. (You can use the **GROUP POOL** rule to define a group pool to be specified as the **TO POOL** target.) See "Policies for automating file management" on page 300.

## Managing storage pools

Managing your storage pools includes:



**Creating storage pools:**

The storage pool to which a disk belongs is an attribute of each disk and is specified as a field in each disk descriptor when the file system is created using the **mmcrfs** command or when disks are added to an existing file system with the **mmadddisk** command. Adding a disk with a new storage pool name in the disk descriptor automatically creates the storage pool.

Storage pool names:

- Must be unique within a file system, but not across file systems.
- Cannot be longer than 255 alphanumeric characters.
- Are case sensitive. **MYpool** and **myPool** are distinct storage pools.

A storage pool is defined by the stanza keyword **pool**; for example:

```
pool=dataPoolA
```

If a storage pool is not specified, the disk is by default assigned to the **system** storage pool.

The **--metadata-block-size** flag on the **mmcrfs** command can be used to create a system pool with a different block size from the user pools. This can be especially beneficial if the default block size is larger than 1 MB. If data and metadata block sizes differ, the system pool must contain only **metadataOnly** disks.

**Changing the storage pool assignment of a disk:**

Once a disk is assigned to a storage pool, the pool assignment cannot be changed by using either the **mmchdisk** command or the **mmrpldisk** command. You can, however, change the pool to which the disk is assigned.

To move a disk to another pool:

1. Delete the disk from its current pool by issuing the **mmdeldisk** command. This will move the data to the remaining disks in the storage pool.
2. Add the disk to the new pool by issuing the **mmadddisk** command.
3. Rebalance the data across all disks in the new storage pool by issuing the **mmrestripefs -P** command.

**Changing the storage pool assignment of a file:**

You can change the storage pool that a file is assigned to.

A root user can change the storage pool that a file is assigned to by either:
- Running **mmapplypolicy** with an appropriate set of policy rules.
- Issuing the **mmchattr -P** command.

By default, both of these commands migrate data immediately (this is the same as using the **-I yes** option for these commands). If desired, you can delay migrating the data by specifying the **-I defer** option for either command. Using the defer option, the existing data does not get moved to the new storage pool until either the **mmrestripefs** command or the **mmrestripefile** command are executed. For additional information, refer to:
- "Overview of policies" on page 300
- "Rebalancing files in a storage pool" on page 299

**Deleting storage pools:**

**System** storage pools, **system.log** pools and user storage pools have different deletion requirements.

Deleting the **System** storage pool is not allowed. You can delete the **system** storage pool only after you have deleted the file system.

You can delete the **system.log** pool by deleting all the disks in the **system.log** pool. You do not need to run a policy to empty the **system.log** pool first, because the **system.log** pool can only contain log files, and those are automatically migrated to the **System** pool when you delete the **System** pool.

In order to delete a user storage pool, you must delete all its disks using the **mmdeldisk** command. When GPFS deletes the last remaining disk from a user storage pool, the storage pool is also deleted. To delete a storage pool, it must be completely empty. A migration policy along with the **mmapplypolicy** command could be used to do this.

**Listing the storage pools of a file system:**

To list the storage pools available for a specific file system, issue the **mmlsfs -P** command.

For example, this command:
```
mmlsfs fs1 -P
```

produces output similar to this:
```
flag                value                    description
------------------- ------------------------ -----------------------------------
 -P                 system;sp1;sp2           Disk storage pools in file system
```

For file system **fs1**, there are three storage pools: the **system** storage pool and user storage pools named **sp1** and **sp2**.

**Listing the storage pool of a file:**

To display the assigned storage pool and the name of the fileset that includes the file, issue the **mmlsattr -L** command.

For example, this command:

```
mmlsattr -L myfile
```

produces output similar to this:

```
file name:            myfile
metadata replication: 2 max 2
data replication:     1 max 2
immutable:            no
appendOnly:           no
flags:
storage pool name:    sp1
fileset name:         root
snapshot name:
creation Time:        Wed Feb 22 15:16:29 2012
Misc attributes:      ARCHIVE
```

File **myfile** is assigned to the storage pool named **sp1** and is part of the root fileset.

**Listing disks and associated statistics:**

To list the disks belonging to a storage pool, issue the **mmdf -P** command.

For example, this command:

```
mmdf fs1 -P sp1
```

produces output similar to this:

```
disk                disk size  failure holds    holds              free KB             free KB
name                    in KB    group metadata data        in full blocks        in fragments
--------------- ------------- -------- -------- ----- -------------------- -------------------
Disks in storage pool: sp1 (Maximum disk size allowed is 1.2 TB)
vp4vsdn05            17760256        6 no       yes        11310080 ( 64%)        205200 ( 1%)
vp5vsdn05            17760256        6 no       yes        11311104 ( 64%)        205136 ( 1%)
vp6vsdn05            17760256        6 no       yes        11300352 ( 64%)        206816 ( 1%)
vp7vsdn05            17760256        6 no       yes        11296256 ( 64%)        209872 ( 1%)
vp0vsdn05            17760256        6 no       yes        11293696 ( 64%)        207968 ( 1%)
vp1vsdn05            17760256        6 no       yes        11293184 ( 64%)        206464 ( 1%)
vp2vsdn05            17760256        6 no       yes        11309056 ( 64%)        203248 ( 1%)
vp3vsdn05            17760256        6 no       yes        11269120 ( 63%)        211456 ( 1%)
             ------------                                -------------------- -------------------
(pool total)        142082048                            90382848 ( 64%)       1656160 ( 1%)
```

This example shows that storage pool **sp1** in file system **fs1** consists of eight disks and identifies details for each disk including:

- Name
- Size
- Failure group
- Data type
- Free space

**Rebalancing files in a storage pool:**

A root user can rebalance file data across all disks in a file system by issuing the **mmrestripefs** command.

Optionally:

- Specifying the **-P** option rebalances only those files assigned to the specified storage pool.
- Specifying the **-p** option rebalances the file placement within the storage pool. For files that are assigned to one storage pool, but that have data in a different pool, (referred to as ill-placed files), using this option migrates their data to the correct pool. (A file becomes "ill-placed" when the **-I defer** option is used during migration of the file between pools.)

**Using replication in a storage pool:**

To enable data replication in a storage pool, you must make certain that there are at least two failure groups within the storage pool.

This is necessary because GPFS maintains separation between storage pools and performs file replication within each storage pool. In other words, a file and its replica must be in the same storage pool. This also means that if you are going to replicate the entire file system, every storage pool in the file system must have at least two failure groups.

**Note:** Depending on the configuration of your file system, if you try to enable file replication in a storage pool having only one failure group, GPFS will either give you a warning or an error message.

## External storage pools

When you initially create a file, GPFS assigns that file to an internal storage pool. Internal storage pools support various types of online storage. To move data from online storage to offline or near-line storage, you can use external storage pools.

External storage pools use a flexible interface driven by GPFS policy rules that simplify data migration to and from other types of storage such as tape storage. For additional information, refer to "Policies for automating file management" on page 300.

You can define multiple external storage pools at any time using GPFS policy rules. To move data to an external storage pool, the GPFS policy engine evaluates the rules that determine which files qualify for transfer to the external pool. From that information, GPFS provides a list of candidate files and executes the script specified in the rule that defines the external pool. That executable script is the interface to the external application, such as IBM Spectrum Protect, that does the actual migration of data into an external pool. Using the external pool interface, GPFS gives you the ability to manage information by allowing you to:

1. Move files and their extended attributes onto low-cost near-line or offline storage when demand for the files diminishes.
2. Recall the files, with all of their previous access information, onto online storage whenever the files are needed.

### External pool requirements

With external pools, GPFS provides metadata processing and the flexibility of using extended file attributes. The external storage manager is responsible for moving files from GPFS and returning them upon the request of an application accessing the file system. Therefore, when you are using external storage pools, you must use an external file management application such as IBM Spectrum Protect. The external application is responsible for maintaining the file once it has left the GPFS file system. For example, GPFS policy rules create a list of files that are eligible for migration. GPFS hands that list to IBM Spectrum Protect which migrates the files to tape and creates a reference file in the file system that has pointers to the tape image. When a file is requested, it is automatically retrieved from the external

storage pool and placed back in an internal storage pool. As an alternative, you can use a GPFS policy rule to retrieve the data in advance of a user request.

The number of external storage pools is only limited by the capabilities of your external application. GPFS allows you to define external storage pools at any time by writing a policy that defines the pool and makes that location known to GPFS. External storage pools are defined by policy rules and initiated by either storage thresholds or use of the **mmapplypolicy** command.

For additional information, refer to "Working with external storage pools" on page 330.

# Policies for automating file management

GPFS provides a means to automate the management of files using policies and rules. Properly managing your files allows you to efficiently use and balance your premium and less expensive storage resources.

GPFS supports these policies:

- *File placement policies* are used to automatically place newly created files in a specific storage pool.
- *File management policies* are used to manage files during their lifecycle by moving them to another storage pool, moving them to near-line storage, copying them to archival storage, changing their replication status, or deleting them.
- *Transparent Cloud Tiering policies* are used to migrate cold data to a cloud storage tier or recall data from the cloud storage tier on reaching certain threshold levels.

You can create and manage policies and policy rules with both the command line interface and the GUI. In the GUI, navigate to **Files** > **Information Lifecycle Management**.

## Overview of policies

A *policy* is a set of rules that describes the life cycle of user data based on the attributes of files. Each rule defines an operation or definition, such as "migrate to a pool and replicate the file."

You can do the following tasks with rules:

- Initial file placement
- File management
- Restoring file data
- Encryption-specific uses. For more information, see the topic *Encryption* in the *IBM Spectrum Scale: Command and Programming Reference*.
- File compression and decompression. For more information, see the topic "File compression" on page 80

When a file is created or restored, the placement policy determines the location of the file's data and assigns the file to a storage pool. All data written to that file is placed in the assigned storage pool.

The placement policy defining the initial placement of newly created files and the rules for placement of restored data must be installed into GPFS with the **mmchpolicy** command. If a GPFS file system does not have a placement policy installed, all the data is stored in the first data storage pool. Only one placement policy can be installed at a time. If you switch from one placement policy to another, or make changes to a placement policy, that action has no effect on existing files. However, newly created files are always placed according to the currently installed placement policy.

The management policy determines file management operations such as migration, deletion, and file compression or decompression.

In order to migrate or delete data, you must use the **mmapplypolicy** command. To compress or decompress data, you can use either the **mmapplypolicy** command with a **MIGRATE** rule or the

**mmchattr** command. You can define the file management rules and install them in the file system together with the placement rules. As an alternative, you may define these rules in a separate file and explicitly provide them to **mmapplypolicy** using the **-P** option. In either case, policy rules for placement or migration may be intermixed. Over the life of the file, data can be migrated to a different storage pool any number of times, and files can be deleted or restored.

**Note:** In a multicluster environment, the scope of the **mmapplypolicy** command is limited to the nodes in the cluster that owns the file system.

**Note:** File compression or decompression using the **mmapplypolicy** command is not supported on the Windows operating system.

File management rules can also be used to control the space utilization of GPFS online storage pools. When the utilization for an online pool exceeds the specified high threshold value, GPFS can be configured, through user exits, to trigger an event that can automatically start **mmapplypolicy** and reduce the utilization of the pool. Using the **mmaddcallback** command, you can specify a script that will run when such an event occurs. For more information, see the topic *mmaddcallback command* in the *IBM Spectrum Scale: Command and Programming Reference*.

GPFS performs error checking for file-placement policies in the following phases:

- When you install a new policy, GPFS checks the basic syntax of all the rules in the policy.
- GPFS also checks all references to storage pools. If a rule in the policy refers to a storage pool that does not exist, the policy is not installed and an error is returned.
- When a new file is created, the rules in the active policy are evaluated in order. If an error is detected, GPFS logs an error, skips all subsequent rules, and returns an **EINVAL** error code to the application.
- Otherwise, the first applicable rule is used to store the file data.

**Default file placement policy:**

When a GPFS file system is first created, the default file placement policy is to assign all files to the **system** storage pool. You can go back to the default policy by running the command:

```
mmchpolicy Device DEFAULT
```

For more information on using GPFS commands to manage policies, see "Managing policies" on page 327.

## Policy rules

A *policy rule* is an SQL-like statement that tells GPFS what to do with the data for a file in a specific storage pool if the file meets specific criteria. A rule can apply to any file being created or only to files being created within a specific fileset or group of filesets.

Rules specify conditions that, when true, cause the rule to be applied. Conditions that cause GPFS to apply a rule include:

- Date and time when the rule is evaluated, that is, the current date and time
- Date and time when the file was last accessed
- Date and time when the file was last modified
- Fileset name
- File name or extension
- File size
- User ID and group ID

GPFS evaluates policy rules in order, from first to last, as they appear in the policy. The first rule that matches determines what is to be done with that file. For example, when a client creates a file, GPFS scans the list of rules in the active file placement policy to determine which rule applies to the file. When

a rule applies to the file, GPFS stops processing the rules and assigns the file to the appropriate storage pool. If no rule applies, an EINVAL error code is returned to the application.

There are eight types of policy rules that allow you to define specific actions that GPFS will implement on the file data. Each rule has clauses that control candidate selection, namely when the rule is allowed to match a file, what files it will match, the order to operate on the matching files and additional attributes to show for each candidate file. Different clauses are permitted on different rules based upon the semantics of the rule.

## Policy rules: Syntax

Policy rules can apply to file placements, group pools, file migrations, file deletions, file exclusions, file lists, file restores, external storage pool definitions, and external list definitions.

The policy rules and their respective syntax diagrams are as follows. For more information about encryption-specific rules, see Chapter 33, "Encryption," on page 537.

- File placement rules

```
RULE ['RuleName']
  SET POOL 'PoolName'
     [LIMIT (OccupancyPercentage)]
     [REPLICATE (DataReplication)]
     [FOR FILESET ('FilesetName'['FilesetName']...)]
     [ACTION (SqlExpression)]
     [WHERE SqlExpression]
```

- Group pool rule; used to define a list of pools that may be used as a pseudo-pool source or destination in either a **FROM POOL** or **TO POOL** clause within another rule

```
RULE ['RuleName'] GROUP POOL ['groupPoolName']
IS 'poolName' [LIMIT(OccupancyPercentage)]
THEN 'poolName2' [LIMIT(n2)]
THEN 'pool-C' [LIMIT(n3)]
THEN ...
```

- File migration rule

```
RULE ['RuleName'] [WHEN TimeBooleanExpression]
  MIGRATE    [COMPRESS ({'yes' | 'no'})]
    [FROM POOL 'FromPoolName']
    [THRESHOLD (HighPercentage[,LowPercentage[,PremigratePercentage]])]
    [WEIGHT (WeightExpression)]
  TO POOL 'ToPoolName'
    [LIMIT (OccupancyPercentage)]
    [REPLICATE (DataReplication)]
    [FOR FILESET ('FilesetName'['FilesetName']...)]
    [SHOW (['String'] SqlExpression)]
    [SIZE (numeric-sql-expression)]
    [ACTION (SqlExpression)]
    [WHERE SqlExpression]
```

  For more information, see the topic "File compression" on page 80.

- File deletion rule

```
RULE ['RuleName'] [WHEN TimeBooleanExpression]
  DELETE
    [DIRECTORIES_PLUS]
    [FROM POOL 'FromPoolName']
    [THRESHOLD (HighPercentage[,LowPercentage])]
    [WEIGHT (WeightExpression)]
    [FOR FILESET ('FilesetName'['FilesetName']...)]
    [SHOW (['String'] SqlExpression)]
    [SIZE (numeric-sql-expression)]
    [ACTION (SqlExpression)]
    [WHERE SqlExpression]
```

- File exclusion rule

```
RULE ['RuleName'] [WHEN TimeBooleanExpression]
  EXCLUDE
   [DIRECTORIES_PLUS]
   [FROM POOL 'FromPoolName']
   [FOR FILESET ('FilesetName'['FilesetName']...)]
   [ACTION (SqlExpression)]
   [WHERE SqlExpression]
```

- File list rule

```
RULE ['RuleName'] [WHEN TimeBooleanExpression]
  LIST 'ListName'
    [EXCLUDE]
    [DIRECTORIES_PLUS]
    [FROM POOL 'FromPoolName']
    [THRESHOLD (HighPercentage[,LowPercentage])]
    [WEIGHT (WeightExpression)]
    [FOR FILESET ('FilesetName'['FilesetName']...)]
    [SHOW (['String'] SqlExpression)]
    [SIZE (numeric-sql-expression)]
    [ACTION (SqlExpression)]
    [WHERE SqlExpression]
```

- File restore rule

```
RULE ['RuleName']
  RESTORE TO POOL 'PoolName'
    [LIMIT (OccupancyPercentage)]
    [REPLICATE (DataReplication)]
    [FOR FILESET ('FilesetName'['FilesetName']...)]
    [ACTION (SqlExpression)]
    [WHERE SqlExpression]
```

- External storage pool definition rule

```
RULE ['RuleName']
  EXTERNAL POOL 'PoolName'
   EXEC 'InterfaceScript'
    [OPTS 'OptionsString ...']
    [ESCAPE '%SpecialCharacters']
    [SIZE sum-number]
```

- External list definition rule

```
RULE ['RuleName']
  EXTERNAL LIST 'ListName'
   EXEC 'InterfaceScript'
    [OPTS 'OptionsString ...']
    [ESCAPE '%SpecialCharacters']
    [THRESHOLD 'ResourceClass']
    [SIZE sum-number]
```

## Policy rules: Terms

The terms of policy rules specify conditions for selecting files and operations to perform on files.

The following terms are used in policy rules. Some terms appear in more than one rule:

**ACTION (***SqlExpression***)**

Specifies an SQL expression that is evaluated only if the other clauses of the rule are satisfied. The action of the *SqlExpression* is completed, and the resulting value of the *SqlExpression* is discarded. In the following example, the rule sets the extended attribute "user.action" to the value "set pool s6" for files that begin with the characters "sp". These files are assigned to the system pool:

```
rule 's6' set pool 'system' action(setxattr('user.action','set pool s6')) where name like 'sp%'
```

**Note:** Encryption policies do not support the **ACTION** clause.

**COMPRESS ({'yes' | 'no})**

Indicates that the file is to be compressed or decompressed. The following rule compresses the files in the pool `datapool` that begin with the string `green%`:

```
RULE 'COMPR1' MIGRATE FROM POOL 'datapool' COMPRESS('yes') WHERE NAME LIKE 'green%'
```

For more information, see the topic "File compression" on page 80.

**DIRECTORIES_PLUS**
Indicates that non-regular file objects (directories, symbolic links, and other items) must be included in the list. If not specified, only ordinary data files are included in the candidate lists.

**DELETE**
Identifies a file deletion rule. A file that matches this rule becomes a candidate for deletion.

**ESCAPE '***%SpecialCharacters***'**
Specifies that path names and the **SHOW**('*string*') expressions within the associated file lists are encoded with a scheme based on RFC3986 URI percent encoding.

Compare the two following rules:

```
RULE 'xp' EXTERNAL POOL 'pool-name' EXEC 'script-name' ESCAPE '%'
RULE 'xl' EXTERNAL LIST 'list-name' EXEC 'script-name' ESCAPE '%/+@#'
```

Both rules specify that all characters except the "unreserved" characters in the set `a-zA-Z0-9-_.~` are encoded as `%XX`, where `XX` comprises two hexadecimal digits.

However, the GPFS **ESCAPE** syntax adds to the set of "unreserved" characters. In the first rule, the syntax `ESCAPE '%'` specifies a rigorous RFC3986 encoding. Under this rule, a path name such as `/root/directory/@abc+def#ghi.jkl` appears in a file list in the following format:

```
%2Froot%2Fdirectory%2F%40abc%2Bdef%23ghi.jkl
```

In the second rule, the syntax `ESCAPE '%/+@#'` specifies that none of the characters in set `/+@#` are escaped. Under this rule, the same path name appears in a file list in the following format:

```
/root/directory/@abc+def#ghi.jkl
```

If you omit the **ESCAPE** clause, the newline character is escaped as `'\n'`, and the backslash character is escaped as `'\\'`; all other characters are presented as is, without further encoding.

**EXCLUDE**
Identifies a file exclusion rule.

**RULE '***x***' EXCLUDE**
A file that matches this form of the rule is excluded from further consideration by any **MIGRATE** or **DELETE** rules that follow.

**RULE '***rule-name***' LIST '***listname-y***' EXCLUDE**
A file that matches this form of the rule is excluded from further consideration by any **LIST** rules that name the same *listname-y*.

**EXEC '***InterfaceScript***'**
Specifies an external program to be invoked to pass requests to an external storage management application. *InterfaceScript* must be a fully qualified path name to a user-provided script or program that supports the commands described in "User-provided program for managing external pools" on page 331.

**EXTERNAL LIST** *ListName*
Defines an external list. This rule does not match files. It provides the binding between the lists that are generated with regular **LIST** rules with a matching *ListName* and the external program that you want to run with these lists as input.

**EXTERNAL POOL** *PoolName*
Defines an external storage pool. This rule does not match files but defines the binding between the policy language and the external storage manager that implements the external storage.

**FOR FILESET ('***FilesetName***'[,'***FilesetName***']...)**
Specifies that the rule applies only to files within the specified filesets.

**FROM POOL** *FromPoolName*
Specifies the name of the source pool from which files are candidates for migration.

**GROUP POOL** *PoolName*
Defines a group pool. This rule supports the concept of distributing data files over several GPFS disk pools.

Optionally, a **LIMIT**, specified as an occupancy percentage, can be specified for each disk pool; if not specified, the limit defaults to 99%. The **THEN** keyword signifies that disk pools that are specified before a **THEN** keyword are preferred over disk pools that are specified after. When a pool that is defined by a **GROUP POOL** rule is the **TO POOL** target of a **MIGRATE** rule, the selected files are distributed among the disk pools that comprise the group pool. Files of highest weight are put into the most preferred disk pool up to the occupancy limit for that pool. If more files must be migrated, they are put into the second most preferred pool up to the occupancy limit for that pool. Again, files of highest weight are selected.

If you specify a file that is defined by a **GROUP POOL** rule in a **FROM POOL** clause, the clause matches any file in any of the disk pools in the group pool.

You can "repack" a group pool by **WEIGHT**. Migrate files of higher weight to preferred disk pools by specifying a group pool as both the source and the target of a **MIGRATE** rule.

```
rule 'grpdef' GROUP POOL 'gpool' IS 'ssd' LIMIT(90) THEN 'fast' LIMIT(85) THEN 'sata'
rule 'repack' MIGRATE FROM POOL 'gpool' TO POOL 'gpool' WEIGHT(FILE_HEAT)
```

See "Tracking file access temperature within a storage pool" on page 295.

**LIMIT (***OccupancyPercentage***)**
Limits the creation of data in a storage pool. GPFS does not migrate a file into a pool if doing so exceeds the occupancy percentage for the pool. If you do not specify an occupancy percentage for a pool, the default value is 99%. See "Phase two: Choosing and scheduling files" on page 321.

You can specify *OccupancyPercentage* as a floating point number, as in the following example:

```
RULE 'r' RESTORE to pool 'x' limit(8.9e1)
```

For testing or planning purposes, and when you use the **mmapplypolicy** command with the **-I defer** or **-I test** option, you can specify a **LIMIT** larger than 100%.

The limit clause does not apply when the target **TO POOL** is a **GROUP POOL**. The limits that are specified in the rule that defines the target **GROUP POOL** govern the action of the **MIGRATE** rule.

**LIST** *ListName*
Identifies a file list generation rule. A file can match more than one list rule but appears in a list only once. *ListName* provides the binding to an **EXTERNAL LIST** rule that specifies the executable program to call when the generated list is processed.

**MIGRATE**
Identifies a file migration rule. A file that matches this rule becomes a candidate for migration to the pool specified by the **TO POOL** clause.

**OPTS '***OptionsString* **...'**
Specifies optional parameters to be passed to the external program defined with the **EXEC** clause. *OptionsString* is not interpreted by the GPFS policy engine.

**REPLICATE (***DataReplication***)**
Overrides the default data replication factor. This value must be specified as 1, 2, or 3.

**RESTORE TO POOL** *PoolName*
Identifies a file restore rule. When you restore a file with the **gpfs_fputattrswithpathname()** subroutine, you can use this rule to match files against their saved attributes rather than the current file attributes. This rule also applies to a command that uses that subroutine, such as the IBM Spectrum Protect command **dsmc restore**.

**RULE ['***RuleName***']**
Initiates the rule statement. *RuleName* identifies the rule and is used in diagnostic messages.

**SET POOL** *PoolName*
Identifies an initial file placement rule. *PoolName* specifies the name of the storage pool where all files that match the rule criteria is placed.

**SHOW (['***String***'] *SqlExpression*)**
Inserts the requested information (the character representation of the evaluated SQL expression *SqlExpression*) into the candidate list that is created by the rule when it deals with external storage pools. *String* is a literal value that gets echoed back.

This clause has no effect in matching files but can be used to define other attributes to be exported with the candidate file lists.

**SIZE (***numeric-sql-expression***)**
Is an optional clause of any **MIGRATE**, **DELETE**, or **LIST** rules that are used for choosing candidate files. *numeric-sql-expression* specifies the size of the file to be used when in calculating the total amount of data to be passed to a user script. The default is **KB_ALLOCATED**.

**SIZE** *sum-number*
Is an optional clause of the **EXTERNAL POOL** and **EXTERNAL LIST** rules. *sum-number* limits the total number of bytes in all of the files named in each list of files passed to your EXEC 'script'. If a single file is larger than *sum-number*, it is passed to your EXEC 'script' as the only entry in a "singleton" file list.

Specify *sum-number* as a numeric constant or a floating-point value.

**Note:** The value of *sum-number* is in kilobytes.

**THRESHOLD (***HighPercentage***[,***LowPercentage***[,***PremigratePercentage***]])**
Controls migration and deletion based on the percent of assigned pool storage that is occupied.

*HighPercentage*
Indicates that the rule is to be applied only if the occupancy percentage of the current pool of the file is greater than or equal to the *HighPercentage* value. Specify a nonnegative integer in the range 0-100.

*LowPercentage*
Indicates that **MIGRATE** and **DELETE** rules are to be applied until the occupancy percentage of the current pool of the file is reduced to less than or equal to the *LowPercentage* value. Specify a nonnegative integer in the range 0-100. The default is 0%.

*PremigratePercentage*
Defines an occupancy percentage of a storage pool that is below the lower limit. Files that lie between the lower limit *LowPercentage* and the pre-migrate limit *PremigratePercentage* are copied and become dual-resident in both the internal GPFS storage pool and the designated external storage pool. This option allows the system to free up space quickly by deleting pre-migrated files if the pool becomes full. Specify a nonnegative integer in the range 0 to *LowPercentage*. The default is the same value as *LowPercentage*.

**Notes:**

1. *Percentage* values can be specified as numeric constants or floating-point values.
2. This option applies only when you migrate to the external storage pool.
3. This option does not apply when the current rule operates on one group pool.

**THRESHOLD (***ResourceClass***)**
Specifies the type of capacity-managed resources that are associated with *ListName*. The following values are valid:

**FILESET_QUOTAS**
Indicates that the **LIST** rule must use the occupancy percentage of the "hard limit" fileset quota per the **mmlsquota** and **mmedquota** commands.

**FILESET_QUOTA_SOFT**
Indicates that the **LIST** rule must use the occupancy percentage of the "soft limit" fileset quota per the **mmlsquota** and **mmedquota** commands.

**GROUP_QUOTAS**
Indicates that the **LIST** rule must use the occupancy percentage of the "hard limit" group quota per the **mmlsquota** and **mmedquota** commands.

**GROUP_QUOTA_SOFT**
Indicates that the **LIST** rule must use the occupancy percentage of the "soft limit" group quota per the **mmlsquota** and **mmedquota** commands.

**POOL_CAPACITIES**
Indicates that the **LIST** rule uses the occupancy percentage of the pool when it applies the threshold rule. This value is the default value. This value is used if the threshold is not specified in the **EXTERNAL LIST** rule but appears in the **LIST** rule.

**USER_QUOTAS**
Indicates that the **LIST** rule uses the occupancy percentage of the "hard limit" user quota per the **mmlsquota** and **mmedquota** commands.

**USER_QUOTA_SOFT**
Indicates that the **LIST** rule uses the occupancy percentage of the "soft limit" user quota per the **mmlsquota** and **mmedquota** commands.

**Note:** This option does not apply when the current rule operates on one group pool.

For more detail on how **THRESHOLD** can be used to control file migration and deletion, see "Phase one: Selecting candidate files" on page 319 and "Pre-migrating files with external storage pools" on page 334.

**TO POOL** *ToPoolName*
Specifies the name of the storage pool to which all the files that match the rule criteria are migrated. This phrase is optional if the **COMPRESS** keyword is specified.

**WEIGHT (***WeightExpression***)**
Establishes an order on the matching files. Specifies an SQL expression with a numeric value that can be converted to a double-precision floating point number. The expression can refer to any of the file attributes and can include any constants and any of the available SQL operators or built-in functions.

**WHEN (***TimeBooleanExpression***)**
Specifies an SQL expression that evaluates to **TRUE** or **FALSE**, depending only on the SQL built-in variable **CURRENT_TIMESTAMP**. If the **WHEN** clause is present and *TimeBooleanExpression* evaluates to **FALSE**, the rule is skipped.

The **mmapplypolicy** command assigns the **CURRENT_TIMESTAMP** when it begins processing. It uses either the actual Coordinated Universal Time date and time or the date specified with the **-D** option.

**WHERE** *SqlExpression*
Specifies an SQL expression that can reference file attributes as SQL variables, functions, and operators. Some attributes are not available to all rules. Compares the file attributes specified in the rule with the attributes of the file that is created.

*SqlExpression* must be an expression that evaluates to **TRUE** or **FALSE**, but can be any combination of standard SQL syntax expressions, including built-in functions.

Omitting the **WHERE** clause entirely is equivalent to writing **WHERE TRUE**. The **WHERE** clause must be the last clause of the rule.

## SQL expressions for policy rules

A number of the available clauses in the GPFS policy rules utilize SQL expressions.

You can reference different file attributes as SQL variables and combine them with SQL functions an operators. Depending on the clause, the SQL expression must evaluate to either **TRUE** or **FALSE**, a numeric value, or a character string. Not all file attributes are available to all rules.

**File attributes in SQL expressions:**

SQL expressions can include file attributes that specify certain clauses.

The following file attributes can be used in SQL expressions specified with the **WHERE**, **WEIGHT**, and **SHOW** clauses:

**ACCESS_TIME**
Specifies an SQL time stamp value for the date and time that the file was last accessed (POSIX atime). See **EXPIRATION_TIME**.

**BLOCKSIZE**
Specifies the size, in bytes, of each block of the file.

**CHANGE_TIME**
Specifies an SQL time stamp value for the date and time that the file metadata was last changed (POSIX ctime).

**CLONE_DEPTH**
Specifies the depth of the clone tree for the file.

**CLONE_IS_PARENT**
Specifies whether the file is a clone parent.

**CLONE_PARENT_FILESETID**
Specifies the fileset ID of the clone parent. The fileset ID is available only if **CLONE_PARENT_IS_SNAP** is a nonzero value.

**CLONE_PARENT_INODE**
Specifies the inode number of the clone parent, or **NULL** if it is not a file clone.

**CLONE_PARENT_IS_SNAP**
Specifies whether the clone parent is in a snapshot.

**CLONE_PARENT_SNAP_ID**
Specifies the snapshot ID of the clone parent. The snapshot ID is available only if **CLONE_PARENT_IS_SNAP** is a nonzero value.

**CREATION_TIME**
Specifies an SQL time stamp value that is assigned when a file is created.

**DEVICE_ID**
Specifies the ID of the device that contains the directory entry.

**DIRECTORY_HASH**
Can be used to group files within the same directory.

**DIRECTORY_HASH** is a function that maps every **PATH_NAME** to a number. All files within the same directory are mapped to the same number and deeper paths are assigned to larger numbers.

**DIRECTORY_HASH** uses the following functions:

**CountSubstr(***BigString*,*LittleString***)**
Counts and returns the number of occurrences of *LittleString* in *BigString*.

**HashToFloat(***StringValue***)**
Is a hash function that returns a quasi-random floating point number ≥ 0 and < 1, whose value

depends on a string value. Although the result might appear random, **HashToFloat**(*StringValue*) always returns the same floating point value for a particular string value.

The following rule lists the directory hash values for three directories:

```
RULE 'y' LIST 'xl' SHOW(DIRECTORY_HASH)
LIST 'xl' /abc/tdir/randy1 SHOW(+3.49449638091027E+000)
LIST 'xl' /abc/tdir/ax     SHOW(+3.49449638091027E+000)
LIST 'xl' /abc/tdir/mmPolicy.8368.765871DF/mm_tmp/PWL.12 SHOW(+5.21282524359412E+000)
LIST 'xl' /abc/tdir/mmPolicy.31559.1E018912/mm_tmp/PWL.3 SHOW(+5.10672733094543E+000)
LIST 'xl' /abc/tdir/mmPolicy.31559.1E018912/mm_tmp/PWL.2 SHOW(+5.10672733094543E+000)
```

The following rule causes files within the same directory to be grouped and processed together during deletion. Grouping the files can improve the performance of GPFS directory-locking and caching.

```
RULE 'purge' DELETE WEIGHT(DIRECTORY_HASH) WHERE (deletion-criteria)
```

**EXPIRATION_TIME**
Specifies the expiration time of the file, expressed as an SQL time-stamp value. If the expiration time of a file is not set, its expiration time is SQL NULL. You can detect such files by checking for "EXPIRATION_TIME IS NULL".

Remember the following points:

- EXPIRATION_TIME is tracked independently from ACCESS_TIME and both values are maintained for immutable files.
- Expiration time and indefinite retention are independent attributes. You can change the value of either one without affecting the value of the other.

**FILE_HEAT**
Specifies the heat of the file based on the file access time and access size. For more information, see `/usr/lpp/mmfs/samples/ilm/README`.

**FILE_SIZE**
Specifies the current size or length of the file, in bytes.

**FILESET_NAME**
Specifies the fileset where the path name for the files is located, or is to be created.

**Note:** Using the **FOR FILESET** clause has the same effect and is more efficient to evaluate.

**GENERATION**
Specifies a number that is incremented whenever an INODE number is reused.

**GROUP_ID**
Specifies the numeric group ID of the file's group.

**GROUP_NAME**
Specifies the group name that is associated with **GROUP_ID**.

**INODE**
Specifies the file's inode number.

**KB_ALLOCATED**
Specifies the number of kilobytes of disk space allocated for the file data.

**MODE**
Displays the type and permission bits of a file as a 10-character string. The string has the same format as the first 10 characters of the output from the UNIX **ls -l** command. For example, **-rwxr-xr-x** is the **MODE** string of a file that can be read or executed by its owner, its group, or any user, but written only by its owner.

The first character of the **MODE** attributes displays the file type. The following values are supported:

**d** Directory.

**l** Link.

**c** Character device.

**b** Block device.

**p** Pipe.

**s** Socket.

**?** Some other attribute, or unknown.

⌂

**MISC_ATTRIBUTES**
Specifies various miscellaneous file attributes. The value is a string of characters that are defined as follows:

| **+** File access is controlled by an Access Control List (ACL).

**a** The file is appendOnly.

**A** Archive.

**c** The file is selected to be compressed.

**D** Directory. To match all directories, you can use %D% as a wildcard character.

| **e** Encrypted. A Microsoft Windows file attribute. Does not refer to GPFS encryption.

| **E** The file has extended-attribute metadata.

| **f** Some data blocks of the file are ill-placed with respect to the File Placement Optimizer (FPO)
| attributes of the file.

**F** Regular data file.

| **H** Hidden. A Microsoft Windows file attribute.

| **i** Not indexed by content. A Microsoft Windows file attribute.

**I** Some data blocks might be ill-placed.

| **j** AFM append flag.

**J** Some data blocks might be ill-replicated.

| **k** AFM remote state flag.

**K** Some data blocks might be ill-compressed.

**L** Symbolic link.

| **m** Empty directory.

**M** Co-managed.

**2** Data blocks are replicated.

**o** Offline.

**O** Other (not F, D, nor L). For example, a device or named pipe.

| **p** Reparse point. A Microsoft Windows file attribute.

| **P** Active File Management (AFM) summary flag. Indicates that at least one specific AFM flag is set:
| **j**, **k**, **u**, **v**, **w**, **x**, **y**, or **z**.

| **r** Has streams. A Microsoft Windows file attribute.

**R** Read-only.

| **s** Sparse. A Microsoft Windows file attribute.

| **S** System. A Microsoft Windows file attribute.

| **t** Temporary. A Microsoft Windows file attribute.

**u** AFM cached-complete flag.

**U** The file is **trunc**-managed.

| **v** AFM create flag.

**V** Read-managed.

| **w** AFM dirty data flag.

**W** Write-managed.

| **x** AFM hard-linked flag.

**X** Immutability.

| **y** AFM attribute-changed flag.

**Y** Indefinite retention.

| **z** AFM local flag.

| **Z** Secure deletion.

**MODIFICATION_SNAPID**
Specifies the integer ID of the snapshot after which the file was last changed. The value is normally derived with the **SNAP_ID()** built-in function that assigns integer values to GPFS snapshot names. This attribute allows policy rules to select files that are modified after a snapshot image is taken.

**MODIFICATION_TIME**
Specifies an SQL time stamp value for the date and time that the file data was last modified (POSIX mtime).

**NAME**
Specifies the name of a file.

**NLINK**
Specifies the number of hard links to the file.

**PATH_NAME**
Specifies a path for the file; the path includes the name of the file.

**POOL_NAME**
Specifies the storage pool where the file data is located.

**Note:** Using the **FROM POOL** clause has the same effect and is often preferable.

**RDEVICE_ID**
Specifies the device type for a device.

**USER_ID**
| Specifies the numeric user ID of the owner of the file. To return the value of USER_ID when
| USER_NAME returns NULL, use COALESCE(USER_NAME, VARCHAR(USER_ID)) .

| **USER_NAME**
| Specifies the user name that is associated with **USER_ID**.

**Notes:**

1. When file attributes are referenced in initial placement rules, only the following attributes are valid: **FILESET_NAME**, **GROUP_ID**, **NAME**, and **USER_ID**. The placement rules, like all rules with a clause, might also reference the current date and current time and use them to control matching.

2. When file attributes are used for restoring files, the attributes correspond to the attributes at the time of the backup, not to the current restored file.
3. For SQL expressions, if you want to show any of these attribute fields as strings (for example, **FILE_HEAT**), use **SHOW('[FILE_HEAT]')** rather than **SHOW('FILE_HEAT')**, as the latter is expanded.
4. All date attributes are evaluated in Coordinated Universal Time (a time standard abbreviated as UTC).

**Using built-in functions:**

With GPFS, you can use built-in functions in comparison predicates, between predicates, in predicates, like predicates, mathematical value expressions, and boolean, string and numeric literals.

These functions are organized into the following categories:

- "Extended attribute functions"
- "String functions" on page 316
- "Numerical functions" on page 317
- "Date and time functions" on page 318

*Extended attribute functions:*

You can use these functions to support access to the extended attributes of a file, and to support conversion of values to the supported SQL data types.

The following attribute functions can be used:

**GetXattrs(***pattern***,***prototype***)**
Returns extended attribute **key=value** pairs of a file for all extended attributes whose keys that match *pattern*. The **key=value** pairs are returned in the format specified by *prototype*.

If the value specified for *pattern* is `'*'` or empty then all keys are matched.

The *prototype* is a character string representing the format of a typical **key=value** pair. The *prototype* allows the user to specify which characters will be used to quote values, escape special code points, separate the key and value, and separate each **key=value** pair.

Some examples of the *prototype* argument include:

```
key~n=value^n,       # specify the escape characters
hexkey=hexvalue,     # specify either or both as hexadecimal values
"key\n"="value\n",   # specify quotes on either or both
key:"value^n";       # specify alternatives to = and ,
k:"v^n";             # allow key and value to be abbreviated
key,                 # specify keys only
"value~n";           # specify values only
key='value~n'&       # alternative quoting character
key=value            # do not use a ',' separator; use space instead
```

You may omit the last or both arguments. The defaults are effectively **GetXattrs('*','key^n=hexvalue,')**.

The **GetXattrs** function returns an empty string for files that have no extended attributes with keys that match *pattern*.

The **GetXattrs** function is supported by the **mmapplypolicy** command, but it might return **NULL** in other contexts.

**SetBGF(***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.

**SetWAD(***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.

**SetWADFG("***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.

Here is an example of using setBGF, setWAD, and setWADFG:

```
RULE 'bgf' SET POOL 'pool1' WHERE NAME LIKE '%' AND setBGF(128) AND setWAD(1) AND setWADFG(1,0,1;2,0,1;3,0,1)
```

This rule has the same effect as the following command:

```
mmchattr --block-group-factor 128 --write-affinity-depth 1 --write-affinity-failuregroup "1,0,1;2,0,1;3,0,1"  test
```

After installing this policy, a newly created file will have the same values for these three extended attributes as it would if **mmchattr** were used to set them:

```
(06:29:11) hs22n42:/sncfs # mmlsattr -L test
file name:            test
metadata replication: 3 max 3
data replication:     3 max 3
immutable:            no
appendOnly:           no
flags:
```

```
storage pool name:    system
fileset name:         root
snapshot name:
Block group factor:  128                                                  -----------------gpfs.BGF
Write affinity depth:  1                                                  -----------------gpfs.WAD
Write Affinity Depth Failure Group(FG) Map for copy:1 1,0,1               -----------------gpfs.WADFG
Write Affinity Depth Failure Group(FG) Map for copy:2 2,0,1
Write Affinity Depth Failure Group(FG) Map for copy:3 3,0,1
creation time:        Sat Jun  8 06:28:50 2013
Misc attributes:      ARCHIVE
```

**SetXattr('***ExtendedAttributeName***', '***ExtendedAttributeValue***')**
This function sets the value of the specified extended attribute of a file.

Successful evaluation of **SetXattr** in a policy rule returns the value **TRUE** and sets the named extended attribute to the specified value for the file that is the subject or object of the rule. This function is effective for policy rules (like **MIGRATE** and **LIST**) that are evaluated by **mmapplypolicy** and for the policy placement rule, **SET POOL**, when a data file is about to be created.

**XATTR(***extended-attribute-name* **[,** *start* **[,** *length***]])**
Returns the value of a substring of the extended attribute that is named by its argument as an SQL VARCHAR value, where:

*extended-attribute-name*
Specifies any SQL expression that evaluates to a character string value. If the named extended attribute does not exist, **XATTR** returns the special SQL value **NULL**.

**Note:** In SQL, the expression **NULL** || *AnyValue* yields **NULL**. In fact, with a few exceptions, the special SQL value of **NULL** "propagates" throughout an SQL expression, to yield **NULL**. A notable exception is that *(expression)* **IS NULL** always yields either **TRUE** or **FALSE**, never **NULL**.

For example, if you wish to display a string like _**NULL**_ when the value of the extended attribute of a file is **NULL** you will need to code your policy rules file like this:

```
define(DISPLAY_NULL,[COALESCE($1,'_NULL_')])
rule external list 'a' exec ''
rule list 'a' SHOW(DISPLAY_NULL(xattr('user.marc')) || ' and ' || DISPLAY_NULL(xattr('user.eric')))
```

Here is an example execution, where either or both of the values of the two named extended attributes may be **NULL**:

```
mmapplypolicy /gig/sill -P /ghome/makaplan/policies/display-null.policy -I test -L 2
   ...
WEIGHT(inf) LIST 'a' /gg/sll/cc SHOW(_NULL_ and _NULL_)
WEIGHT(inf) LIST 'a' /gg/sll/mm SHOW(yes-marc and _NULL_)
WEIGHT(inf) LIST 'a' /gg/sll/bb SHOW(_NULL_ and yes-eric)
WEIGHT(inf) LIST 'a' /gg/sll/tt SHOW(yes-marc and yes-eric)
```

GPFS/Policy/SQL is a subset of standard ISO/ANSI SQL, with additional extensions and modifications to facilitate GPFS/ILM. Regarding the **NULL** value, GPFS/Policy/SQL supports the "unknown value" meaning of **NULL**.

*start*
Is the optional starting position within the extended attribute value. The default is 1.

*length*
Is the optional length, in bytes, of the extended attribute value to return. The default is the number of bytes from the start to the end of the extended attribute string.

**Note:** **XATTR** (*name,i,k*) == **SUBSTR**(**XATTR**(*name*),*i,k*).

Some extended attribute values represent numbers or timestamps as decimal or binary strings. Use the **TIMESTAMP**, **XATTR_FLOAT**, or **XATTR_INTEGER** function to convert extended attributes to SQL numeric or timestamp values:

**XATTR_FLOAT(***extended-attribute-name* **[,** *start* **[,** *length***, [,** *conversion_option***]]])**
Returns the value of a substring of the extended attribute that is named by its argument, converted to an SQL double floating-point value, where:

*extended-attribute-name*
Specifies any SQL expression that evaluates to a character string value. If the named extended attribute does not exist, **XATTR** returns the special SQL value **NULL**.

*start*
Is the optional starting position within the extended attribute value. The default is 1.

*length*
Is the optional length, in bytes, of the extended attribute value to return. The default is the number of bytes from the start to the end of the extended attribute string. You can specify length as -1 to reach from the start to the end of the extended attribute string.

*conversion_option*
Specifies how the bytes are to be converted to a floating-point value. Supported options include:

- BIG_ENDIAN_DOUBLE or BD - a signed binary representation, IEEE floating, sign + 11 bit exponent + fraction. This is the default when executing on a "big endian" host OS, such as AIX on PowerPC®.
- BIG_ENDIAN_SINGLE or BS - IEEE floating, sign + 8-bit exponent + fraction.
- LITTLE_ENDIAN_DOUBLE or LD - bytewise reversed binary representation. This is the default when executing on a "little endian" host OS, such as Linux on Intel x86.
- LITTLE_ENDIAN_SINGLE or LS - bytewise-reversed binary representation.
- DECIMAL - the conventional SQL character string representation of a floating-point value.

**Notes:**

1. Any prefix of a conversion name can be specified instead of spelling out the whole name. The first match against the list of supported options is used; for example, L matches LITTLE_ENDIAN_DOUBLE.
2. If the extended attribute does not exist, the selected substring has a length of 0, or the selected bytes cannot be converted to a floating-point value, the function returns the special SQL value **NULL**.

**XATTR_INTEGER(***extended-attribute-name* **[,** *start* **[,** *length***, [,** *conversion_option***]]])**
Returns the value of (a substring of) the extended attribute named by its argument, converted to a SQL LARGEINT value, where.

*extended-attribute-name*
Specifies any SQL expression that evaluates to a character string value. If the named extended attribute does not exist, **XATTR** returns the special SQL value **NULL**.

*start*
Is the optional starting position within the extended attribute value. The default is 1.

*length*
Is the optional length, in bytes, of the extended attribute value to return. The default is the number of bytes from the start to the end of the extended attribute string. You can specify length as -1 to reach from the start to the end of the extended attribute string.

*conversion_option*
Specifies how the bytes are to be converted to a LARGEINT value. Supported options include:

- BIG_ENDIAN - a signed binary representation, most significant byte first. This is the default when executing on a "big endian" host OS, such as AIX on PowerPC.

- LITTLE_ENDIAN - bytewise reversed binary representation. This is the default when executing on a "little endian" host OS, such as Linux on Intel x86.
- DECIMAL - the conventional SQL character string representation of an integer value.

**Notes:**

1. Any prefix of a conversion name can be specified instead of spelling out the whole name (B, L, or D, for example).
2. If the extended attribute does not exist, the selected substring has a length of 0, or the selected bytes cannot be converted to a LARGEINT value, the function returns the special SQL value **NULL**. For example:

```
XATTR_INTEGER('xyz.jim',5,-1,'DECIMAL')
```

*String functions:*

You can use these string-manipulation functions on file names and literal values.

**Important tips:**

1. You must enclose strings in single-quotation marks.
2. You can include a single-quotation mark in a string by using two single-quotation marks. For example, **'a''b'** represents the string **a'b**.

**CHAR(***expr***[, *length*])**
Returns a fixed-length character string representation of its *expr* argument, where:

*expr*
Can be any data type.

*length*
If present, must be a literal, integer value.

The resulting type is **CHAR** or **VARCHAR**, depending upon the particular function called.

The string that **CHAR** returns is padded with blanks to fill the *length* of the string. If *length* is not specified, it defaults to a value that depends on the type of the argument (*expr*).

**Note:** The maximum length of a **CHAR** (fixed length string) value is 255 bytes. The result of evaluating an SQL expression whose result is type **CHAR** may be truncated to this maximum length.

**CONCAT(***x***,***y***)**
Concatenates strings *x* and *y*.

**HEX(***x***)**
Converts an integer *x* into hexadecimal format.

**LENGTH(***x***)**
Determines the length of the data type of string *x*.

**LOWER(***x***)**
Converts string *x* into lowercase.

**REGEX(***String***,'***Pattern***')**
Returns **TRUE** if the pattern matches, **FALSE** if it does not. *Pattern* is a Posix extended regular expression.

**Note:** The policy SQL parser normally performs M4 macro preprocessing with square brackets set as the quote characters. Therefore, it is recommended that you add an extra set of square brackets around your **REGEX** pattern string; for example:

```
...WHERE REGEX(name,['^[a-z]*$'])  /* only accept lowercase alphabetic file names */
```

The following SQL expression:

```
NOT REGEX(STRING_VALUE,['^[^z]*$|^[^y]*$|^[^x]*$|[abc]'])
```

can be used to test if STRING_VALUE contains *all* of the characters x, y, and z, in any order, and *none* of the characters a, b, or c.

**REGEXREPLACE(***string***,***pattern***,***result-prototype-string***)**
Returns a character string as *result-prototype-string* with occurrences of *i* (where *i* is 0 through 9) replaced by the substrings of the original string that match the $i^{th}$ parenthesis delimited parts of the pattern string. For example:

```
REGEXREPLACE('speechless',['([^aeiou]*)([aeiou]*)(.*)'],['last=\3. middle=\2. first=\1.'])
```

returns the following:

```
'last=chless. middle=ee. first=sp.'
```

When *pattern* does not match *string*, **REGEXREPLACE** returns the value **NULL**.

When a **\0** is specified in the *result-prototype-string*, it is replaced by the substring of *string* that matches the entire *pattern*.

**SUBSTR(***x***,***y***,***z***)**
Extracts a portion of string *x*, starting at position *y*, optionally for *z* characters (otherwise to the end of the string). This is the short form of **SUBSTRING**. If *y* is a negative number, the starting position is counted from the end of the string; for example, **SUBSTR('ABCDEFGH',-3,2)** == 'FG'.

**Note:** Do not confuse **SUBSTR** with **substr**. **substr** is an m4 built-in macro function.

**SUBSTRING(***x* **FROM** *y* **FOR** *z***)**
Extracts a portion of string *x*, starting at position *y*, optionally for *z* characters (otherwise to the end of the string).

**UPPER(***x***)**
Converts the string *x* into uppercase.

**VARCHAR(***expr* **[,** *length* **])**
Returns a varying-length character string representation of a character string, date/time value, or numeric value, where:

*expr*
Can be any data type.

*length*
If present, must be a literal, integer value.

The resulting type is **CHAR** or **VARCHAR**, depending upon the particular function called. Unlike **CHAR**, the string that the **VARCHAR** function returns is not padded with blanks.

**Note:** The maximum length of a **VARCHAR**(variable length string) value is 8192 bytes. The result of evaluating an SQL expression whose result is type **VARCHAR** may be truncated to this maximum length.

*Numerical functions:*

You can use numeric-calculation functions to place files based on either numeric parts of the file name, numeric parts of the current date, or UNIX-client user IDs or group IDs.

These functions can be used in combination with comparison predicates and mathematical infix operators (such as addition, subtraction, multiplication, division, modulo division, and exponentiation).

**INT(***x***)**
Converts number *x* to a whole number, rounding up fractions of .5 or greater.

**INTEGER(*x*)**
Converts number *x* to a whole number, rounding up fractions of .5 or greater.

**MOD(*x*,*y*)**
Determines the value of *x* taken modulo *y* (*x* % *y*).

*Date and time functions:*

You can use these date-manipulation and time-manipulation functions to place files based on when the files are created and the local time of the GPFS node serving the directory where the file is being created.

**CURRENT_DATE**
Determines the current date on the GPFS server.

**CURRENT_TIMESTAMP**
Determines the current date and time on the GPFS server.

**DAYOFWEEK(*x*)**
Determines the day of the week from date or timestamp *x*. The day of a week is from 1 to 7 (Sunday is 1).

**DAYOFYEAR(*x*)**
Determines the day of the year from date *x*. The day of a year is a number from 1 to 366.

**DAY(*x*)**
Determines the day of the month from date or timestamp *x*.

**DAYS(*x*)**
Determines the number of days between date or timestamp *x* and 0001-01-01.

**DAYSINMONTH(*x*)**
Determines the number of days in the month of date *x*.

**DAYSINYEAR(*x*)**
Determines the day of the year of date *x*.

**HOUR(*x*)**
Determines the hour of the day (a value from 0 to 23) of timestamp *x*.

**MINUTE(*x*)**
Determines the minute from timestamp *x*.

**MONTH(*x*)**
Determines the month of the year from date or timestamp *x*.

**QUARTER(*x*)**
Determines the quarter of year from date *x*. Quarter values are the numbers 1 through 4. For example, January, February, and March are in quarter 1.

**SECOND(*x*)**
Returns the seconds portion of timestamp *x*.

**TIMESTAMP(*sql-numeric-value*) or TIMESTAMP(*sql-character-string-value*)**
Accepts any numeric value. The numeric value is interpreted as the number of seconds since January 1, 1970 (the standard UNIX epoch) and is converted to an SQL TIMESTAMP value.

Signed 64-bit LARGEINT argument values are supported. Negative argument values cause **TIMESTAMP** to convert these values to timestamps that represent years before the UNIX epoch.

This function also accepts character strings of the form *YYYY-MM-DD HH:MM:SS*. A hyphen (-) or an at sign (@) might appear instead of the blank between the date and the time. The time can be omitted. An omitted time defaults to 00:00:00. The *:SS* field can be omitted, which defaults to 00.

**WEEK(*x*)**
Determines the week of the year from date *x*.

**YEAR(***x***)**
Determines the year from date or timestamp *x*.

All date and time functions use Universal Time (UT).

**Example of a policy rules file**

```
/*
Sample GPFS policy rules file
*/

rule 'vip' set pool 'pool0' where USER_ID <= 100
RULE 'm1' SET POOL 'pool1' WHERE LOWER(NAME) LIKE '%marc%'
RULE SET POOL 'pool1' REPLICATE (2) WHERE UPPER(NAME) = '%IBM%'
RULE 'r2' SET POOL 'pool2' WHERE UPPER(SUBSTRING(NAME FROM 1 FOR 4)) = 'GPFS'
RULE 'r3' SET POOL 'pool3' WHERE LOWER(SUBSTR(NAME,1,5)) = 'roger'
RULE SET POOL 'pool4' WHERE LENGTH(NAME) = 7
RULE SET POOL 'pool5' WHERE name like 'xyz%' AND name like '%qed' OR name like '%.tmp%'
RULE SET POOL 'pool6' WHERE name like 'abc%' OR name like '%xyz' AND name like 'x%'

RULE 'restore' RESTORE TO POOL 'pool0' where USER_ID <= 100


/* If none of the rules matches put those files in system pool */
rule 'default' SET POOL 'system'
```

*Miscellaneous SQL functions:*

The following miscellaneous SQL functions are available.

**SNAP_ID(['***FilesetName***',] '***SnapshotName***')**
Given an (optional) fileset/inode-space name and a snapshot name, this function returns the numeric snapshot ID of the given snapshot within the given inode-space.

**GetEnv('***EnvironmentVariableName***')**
This function gets the value of the specified environment variable.

**GetMMconfig('***GPFSConfigurationParameter***')**
This function gets the value of the specified GPFS configuration parameter.

# The mmapplypolicy command and policy rules

The **mmapplypolicy** command has policy rules that are based on the characteristics of different phases.

Any given file is a potential candidate for at most one **MIGRATE** or **DELETE** operation during each invocation of the **mmapplypolicy** command. A single invocation of the **mmapplypolicy** command is called the *job*.

The **mmapplypolicy** command sets the SQL built-in variable **CURRENT_TIMESTAMP**, and collects pool occupancy statistics at the beginning of the job.

The **mmapplypolicy** job consists of three major phases:

1. "Phase one: Selecting candidate files"
2. "Phase two: Choosing and scheduling files" on page 321
3. "Phase three: Migrating and deleting files" on page 322

## Phase one: Selecting candidate files

In the first phase of the **mmapplypolicy** job, all the files within the specified GPFS file system device, or below the input path name, are scanned. The attributes of each file are read from the file's GPFS inode structure.

**Note: mmapplypolicy** reads directly from the metadata disk blocks and can therefore lag behind the posix state of the file system. To be sure that `MODIFICATION_TIME` and the other timestamps are completely up to date, you can use the following suspend-and-resume sequence to force recent changes to disk:

```
mmfsctl fs-name suspend; mmfsctl fs-name resume;
```

For each file, the policy rules are considered, in order, from first rule to last:

- If the rule has a **WHEN** clause that evaluates to **FALSE**, the rule is skipped.
- If the rule has a **FROM POOL** clause, and the named pool does not match the **POOL_NAME** attribute of the file, the rule is skipped. A **FROM POOL** clause that specifies a group pool name matches a file if any pool name within the group pool matches the **POOL_NAME** attribute of the file.
- If there is a **THRESHOLD** clause and the current pool of the file has an occupancy percentage that is less than the *HighPercentage* parameter of the **THRESHOLD** clause, the rule is skipped.
- If the rule has a **FOR FILESET** clause, but none of the named filesets match the **FILESET_NAME** attribute of the file, the rule is skipped.
- If the rule has a **WHERE** clause that evaluates to **FALSE**, the rule is skipped. Otherwise, the rule applies.
- If the applicable rule is a **LIST** '*listname-y*' rule, the file becomes a candidate for inclusion in the named list unless the **EXCLUDE** keyword is present, in which case the file will not be a candidate; nor will any following **LIST** '*listname-y*' rules be considered for the subject file. However, the file is subject to **LIST** rules naming other list names.
- If the applicable rule is an **EXCLUDE** rule, the file will be neither migrated nor deleted. Files matching the **EXCLUDE** rule are not candidates for any **MIGRATE** or **DELETE** rule.

  **Note:** Specify the **EXCLUDE** rule before any other rules that might match the files that are being excluded. For example:

  ```
  RULE 'Exclude root's file' EXCLUDE where USER_ID = 0
  RULE 'Migrate all but root's files' MIGRATE TO POOL 'pool1'
  ```

  will migrate all the files that are not owned by **root**. If the **MIGRATE** rule was placed in the policy file before the **EXCLUDE** rule, all files would be migrated because the policy engine would evaluate the rules from first to last, and **root**'s files would have to match the **MIGRATE** rule.

  To exclude files from matching a **LIST** rule, you must create a separate **LIST** rule with the **EXCLUDE** clause and place it before the **LIST** rule.
- If the applicable rule is a **MIGRATE** rule, the file becomes a *candidate* for migration to the pool specified by the **TO POOL** clause.

  When a group pool is the **TO POOL** target of a **MIGRATE** rule, the selected files are distributed among the disk pools comprising the group pool, with files of highest weight going to the most preferred disk pool up to the occupancy limit for that pool. If there are still more files to be migrated, those go to the second most-preferred pool up to the occupancy limit for that pool (again choosing the highest-weight files from among the remaining selected files); and so on for the subsequent most-preferred pools, until either all selected files have been migrated or until all the disk pools of the group pool have been filled to their respective limits.
- If the applicable rule is a **DELETE** rule, the file becomes a *candidate* for deletion.
- If there is no applicable rule, the file is not a candidate for migration or deletion.
- Each candidate file (for migration or deletion) is also associated with a *LowPercentage* occupancy percentage value, which is taken from the **THRESHOLD** clause of the applicable rule. If not specified, the *LowPercentage* value defaults to 0%.
- Each candidate file is also associated with a numeric *weight*, either computed from the *WeightExpression* of the applicable rule, or assigned a default using these rules:
  - If a *LowPercentage* is specified within a **THRESHOLD** clause of the applicable rule, the weight of the candidate is taken as the **KB_ALLOCATED** attribute of the candidate file.

– If a *LowPercentage* is not specified within a **THRESHOLD** clause of the applicable rule, the weight of the candidate is taken as **+infinity**.

## Phase two: Choosing and scheduling files

In the second phase of the **mmapplypolicy** job, some or all of the candidate files are chosen.

Chosen files are scheduled for migration or deletion, taking into account the weights and thresholds determined in "Phase one: Selecting candidate files" on page 319, as well as the actual pool occupancy percentages. Generally, candidates with higher weights are chosen ahead of those with lower weights.

File migrations to and from external pools are done before migrations and deletions that involve only GPFS disk pools.

File migrations that do not target group pools are done before file migrations to group pools.

File migrations that target a group pool are done so that candidate files with higher weights are migrated to the more preferred GPFS disk pools within the group pool, but respecting the **LIMIT**s specified in the group pool definition.

The following two options can be used to adjust the method by which candidates are chosen:

**--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.

**--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.

### File grouping and the SIZE clause

When scheduling files, **mmapplypolicy** simply groups together either the next 100 files by default, or the number of files explicitly set using the **-B** option.

However, you can set up **mmapplypolicy** to schedule files so that each invocation of the InterfaceScript gets approximately the same amount of file data to process. To do so, use the **SIZE** clause of certain

policy rules to specify that scheduling be based on the sum of the sizes of the files. The **SIZE** clause can be applied to the following rules (for details, see "Policy rules" on page 301):

- **DELETE**
- **EXTERNAL LIST**
- **EXTERNAL POOL**
- **LIST**
- **MIGRATE**

### Administrator-specified customized file grouping or aggregation

In addition to using the **SIZE** clause to control the *amount* of work passed to each invocation of a InterfaceScript, you can also specify that files with *similar attributes* be grouped or aggregated together during the scheduling phase. To do so, use an aggregator program to take a list of chosen candidate files, sort them according to certain attributes, and produce a reordered file list that can be passed as input to the user script.

You can accomplish this by following these steps:

1. Run **mmapplypolicy** with the **-I prepare** option to produce a list of chosen candidate files, but not pass the list to a InterfaceScript.
2. Use your aggregator program to sort the list of chosen candidate files into groups with similar attributes and write each group to a new, separate file list.
3. Run **mmapplypolicy** with the **-r** option, specifying a set of file list files to be read. When invoked with the **-r** option, **mmapplypolicy** does not choose candidate files; rather, it passes the specified file lists as input to the InterfaceScript.

   **Note:** You can also use the **-q** option to specify that small groups of files are to be taken in round-robin fashion from the input file lists (for example, take a small group of files from x.list.A, then from x.list.B, then from x.list.C, then back to x.list.A, and so on, until all of the files have been processed).

   To prevent **mmapplypolicy** from redistributing the grouped files according to size, omit the **SIZE** clause from the appropriate policy rules and set the bunching parameter of the **-B** option to a very large value.

### Reasons for candidates not to be chosen for deletion or migration

Generally, a candidate is not chosen for deletion from a pool, nor migration out of a pool, when the pool occupancy percentage falls below the *LowPercentage* value. Also, candidate files will not be chosen for migration into a target **TO POOL** when the target pool reaches the occupancy percentage specified by the **LIMIT** clause (or 99% if no **LIMIT** was explicitly specified by the applicable rule).

The limit clause does not apply when the target **TO POOL** is a group pool; the limits specified in the rule defining the target group pool govern the action of the **MIGRATE** rule. The policy-interpreting program (for example, **mmapplypolicy**) may issue a warning if a **LIMIT** clause appears in a rule whose target pool is a group pool.

## Phase three: Migrating and deleting files

In the third phase of the **mmapplypolicy** job, the candidate files that were chosen and scheduled by the second phase are migrated or deleted, each according to its applicable rule.

For migrations, if the applicable rule had a **REPLICATE** clause, the replication factors are also adjusted accordingly. It is acceptable for the effective **FROM POOL** and **TO POOL** to be the same because the **mmapplypolicy** command can be used to adjust the replication factors of files without necessarily moving them from one pool to another.

The migration performed in the third phase can involve large amounts of data movement. Therefore, you may want to consider using the **–I defer** option of the **mmapplypolicy** command, and then perform the data movements with the **mmrestripefs -p** command.

## Policy rules: Examples and tips

Before you write and apply policies, consider the following advice.

You are advised to test your rules using the **mmapplypolicy** command with the **-I test** option and the **-L 3** (or higher) option. This will help you understand which files are selected as candidates, and which candidates are chosen.

Do not apply a policy to an entire file system of vital files until you are confident that the rules correctly express your intentions. To test your rules, find or create a subdirectory with a modest number of files, some that you expect to be selected by your SQL policy rules and some that you expect will be skipped.

Then run the following command:

```
mmapplypolicy /TestSubdirectory  -L 6  -I test
```

The output will show you exactly which files are scanned, and which match rules or no rules. If a problem is not apparent, you can add a SHOW() clause to your rule or rules to examine the values of the file attributes of interest or the value of any SQL expression. To examine several, use the following:

```
SHOW('x1=' || varchar(Expression1) || ' x2=' || varchar(Expression2) || ... )
```

where *ExpressionX* is the SQL variable or expression of function that you suspect or do not understand. Beware that if any expression evaluates to SQL NULL, the entire show clause will be NULL, by the rules of SQL. One way to show null vs. non-null values is to define a macro and use it as in the following example:

```
define(DISPLAY_NULL,[CASE WHEN ($1) IS NULL THEN '_NULL_' ELSE varchar($1) END])

rule list a SHOW( 'x1=' || DISPLAY_NULL(xattr('user.marc')) || ' and  x2=' || DISPLAY_NULL(xattr('user.eric')))
```

**Note:** For examples and more information on the **-L** flag, see the topic *The mmapplypolicy -L command* in the *IBM Spectrum Scale: Problem Determination Guide*.

1. Delete files from the storage pool named **pool_1** that have not been accessed in the last 30 days, and are named like temporary files or appear in any directory that is named **tmp**:

   ```
   RULE 'del1' DELETE FROM POOL 'pool_1'
      WHERE (DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME) > 30)
           AND (lower(NAME) LIKE '%.tmp' OR PATH_NAME LIKE '%/tmp/%')
   ```

2. Use the SQL **LIKE** predicate to test file names and path names:

   ```
   RULE '*/_*'  DELETE WHERE PATH_NAME LIKE '%/x_%' ESCAPE 'x'
   RULE '*XYZ*' DELETE WHERE NAME_LIKE '%XYZ%'
   RULE '12_45' DELETE WHERE NAME LIKE '12x_45' ESCAPE 'x'
   RULE '12%45' DELETE WHERE NAME LIKE '12x%45' ESCAPE 'x'
   RULE '12?45' DELETE WHERE NAME LIKE '12_45'
   RULE '12*45' DELETE WHERE NAME LIKE '12%45'
   RULE '*_*'   DELETE WHERE NAME LIKE '%x_%' ESCAPE 'x'
   ```

   Where:

   - A percent % wildcard in the name represents zero or more characters.
   - An underscore (_) wildcard in the name represents one byte.

   Use the optional **ESCAPE** clause to establish an escape character, when you need to match '_' or '%' exactly.

3. Use the SQL **UPPER** and **LOWER** functions to ignore case when testing names:

   ```
   RULE 'UPPER'  DELETE WHERE upper(PATH_NAME) LIKE '%/TMP/OLD/%'
   RULE 'lower'  DELETE WHERE lower(PATH_NAME) LIKE '%/tmp/old/%'
   ```

4. Use the SQL **SUBSTR** or **SUBSTRING** functions to test a substring of a name:

```
RULE 's1' DELETE WHERE SUBSTRING(NAME FROM 1 FOR 5)='XXXX-'
RULE 's2' DELETE WHERE SUBSTR(NAME,1,5)='YYYY-'
```

5. Use the SQL **SUBSTR** and **LENGTH** functions to test the suffix of a name:

```
RULE 'sfx' DELETE WHERE SUBSTR(NAME,LENGTH(NAME)-3)='.tmp'
```

6. Use a **WHEN** clause to restrict rule applicability to a particular day of the week:

```
RULE 'D_SUN' WHEN (DayOfWeek(CURRENT_DATE)=1) /* Sunday */
  DELETE WHERE PATH_NAME LIKE '%/tmp/%'
```

**CURRENT_DATE** is an SQL built in operand that returns the date portion of the **CURRENT_TIMESTAMP** value.

7. Use the SQL **IN** operator to test several possibilities:

```
RULE 'D_WEEKEND' WHEN (DayOfWeek(CURRENT_DATE) IN (7,1)) /*  Saturday or Sunday */
  DELETE WHERE PATH_NAME LIKE '%/tmp/%'
```

For information on how to use a macro processor such as **m4** to make reading and writing policy rules easier, see "Using macro processing utilities with policy rules" on page 326.

8. Use a **FILESET** clause to restrict the rule to files within particular filesets:

```
RULE 'fsrule1' MIGRATE TO POOL 'pool_2'
         FOR FILESET('root','fset1')
```

In this example there is no **FROM POOL** clause, so regardless of their current storage pool placement, all files from the named filesets are subject to migration to storage pool **pool_2**.

9. Use an **EXCLUDE** rule to exclude a set of files from all subsequent rules:

```
RULE 'Xsuper' EXCLUDE WHERE USER_ID=0
RULE 'mpg'    DELETE  WHERE lower(NAME) LIKE '%.mpg' AND FILE_SIZE>20123456
```

**Notes:**

a. Specify the **EXCLUDE** rule before rules that might match the files that are being excluded.

b. You cannot define a list and what to exclude from the list in a single rule. You must define two LIST statements, one specifying which files will be in the list, and one specifying what to exclude from the list. For example, to exclude files containing the word **test** from the LIST rule **allfiles**, define the following:

```
RULE EXTERNAL LIST 'allfiles' EXEC '/u/brownap/policy/CHE/exec.list'

RULE 'exclude_allfiles' LIST 'allfiles' EXCLUDE where name like '%test%'

RULE 'all' LIST 'allfiles' SHOW('misc_attr ='|| MISC_ATTRIBUTES || HEX(MISC_ATTRIBUTES)) \
  where name like '%'
```

10. Use the SQL **NOT** operator with keywords, along with **AND** and **OR**:

```
RULE 'D_WEEKDAY' WHEN (DayOfWeek(CURRENT_DATE) NOT IN (7,1)) /*  a weekday */
  DELETE WHERE (PATH_NAME LIKE '%/tmp/%' OR NAME LIKE '%.tmp')
          AND (KB_ALLOCATED > 9999 AND NOT USER_ID=0)
```

11. Use a **REPLICATE** clause to increase the availability of selected files:

```
RULE 'R2' MIGRATE FROM POOL 'ypooly' TO POOL 'ypooly'
  REPLICATE(2) WHERE USER_ID=0
```

Before increasing the data replication factor for any file, the file system must be configured to support data replication.

12. The difference of two SQL Timestamp values may be compared to an SQL Interval value:

```
rule 'a' migrate to pool 'A' where CURRENT_TIMESTAMP - MODIFICATION_TIME > INTERVAL  '10' DAYS
rule 'b' migrate to pool 'B' where CURRENT_TIMESTAMP - MODIFICATION_TIME > INTERVAL  '10' HOURS
rule 'c' migrate to pool 'C' where CURRENT_TIMESTAMP - MODIFICATION_TIME > INTERVAL  '10' MINUTES
rule 'd' migrate to pool 'D' where CURRENT_TIMESTAMP - MODIFICATION_TIME > INTERVAL  '10' SECONDS
```

For the best precision, use the `INTERVAL...SECONDS` construct.

13. By carefully assigning both weights and thresholds, the administrator can formally express rules like this:

If the storage pool named **pool_X** has an occupancy percentage above 90% now, bring the occupancy percentage of storage pool named **pool_X** down to 80% by migrating files that are three months or older to the storage pool named **pool_ZZ**. But, if you can find enough year-old files to bring the occupancy percentage down to 50%, do that also.

```
RULE 'year-old' MIGRATE FROM POOL 'pool_X'
   THRESHOLD(90,50) WEIGHT(weight_expression)
   TO POOL 'pool_ZZ'
   WHERE  DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME) > 365

RULE '3month-old' MIGRATE FROM POOL 'pool_X'
   THRESHOLD(90,80) WEIGHT(weight_expression)
   TO POOL 'pool_ZZ'
   WHERE  DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME) > 90
```

More information about weights is available in the next example.

A goal of this **mmapplypolicy** job is to reduce the occupancy percentage of the **FROM POOL** to the low occupancy percentage specified on the **THRESHOLD** clause, if possible. The **mmapplypolicy** job does not migrate or delete more files than are necessary to produce this occupancy percentage. The task consists of these steps:

a. Each candidate file is assigned a weight.
b. All candidate files are sorted by weight.
c. The highest weight files are chosen to **MIGRATE** or **DELETE** until the low occupancy percentage is achieved, or there are no more candidates.

The administrator who writes the rules must ensure that the computed weights are as intended, and that the comparisons are meaningful. This is similar to the IBM Spectrum Protect convention, where the weighting function for each file is determined by the equation:

```
X * access_age + Y * file_size
```

where:

access_age is **DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME)**

file_size is **FILE_SIZE** or **KB_ALLOCATED**

**X and Y are weight factors chosen by the system administrator.**

14. The **WEIGHT** clause can be used to express ideas like this (stated informally):

```
IF access_age > 365 days THEN weight = 100000 + access_age
ELSE IF access_age < 30 days THEN weight = 0
ELSE weight= KB_ALLOCATED
```

This means:

- Give a very large weight bias to any file older than a year.
- Force the weight of any file younger than 30 days to 0.
- Assign weights to all other files according to the number of kilobytes occupied.

The formal SQL syntax is this:

```
CASE
 WHEN DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME) > 365
  THEN 100000 + DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME)
 WHEN DAYS(CURRENT_TIMESTAMP) – DAYS(ACCESS_TIME) < 30
  THEN 0
 ELSE
  KB_ALLOCATED
END
```

15. The **SHOW** clause has no effect in matching files but can be used to define additional attributes to be exported with the candidate file lists. It may be used for any purpose but is primarily used to support file aggregation.

To support aggregation, you can use the **SHOW** clause to output an aggregation value for each file selected by a rule. You can then output those values to a file list and input that list to an external program that groups the files into aggregates.

16. If you have a large number of filesets against which to test, use the **FILESET_NAME** variable as shown in the following example:

```
RULE 'x' SET POOL 'gold' WHERE FILESET_NAME LIKE 'xyz.%.xyz'
```

However, if you are testing against just a few filesets, you can use the `FOR FILESET('xyz1', 'xyz2')` form instead.

17. You may convert a time interval value to a number of seconds using SQL cast syntax; for example:

```
define([toSeconds],[(($1) SECONDS(12,6))])

define([toUnixSeconds],[toSeconds($1 - '1970-1-1@0:00')])

RULE external list b
RULE list b SHOW('sinceNow=' toSeconds(current_timestamp-modification_time) )
RULE external list c
RULE list c SHOW('sinceUnixEpoch=' toUnixSeconds(modification_time) )
```

The following is also supported:

```
define(access_age_in_days,( INTEGER(( (CURRENT_TIMESTAMP - ACCESS_TIME) SECONDS)) /(24*3600.0) ) )

RULE external list w exec ''
RULE list w weight(access_age_in_days) show(access_age_in_days)
```

## Using macro processing utilities with policy rules

Prior to evaluating the policy rules, GPFS invokes the **m4** macro processor to process the policy file.

This processing allows you to incorporate into the policy file some of the traditional **m4** facilities and to define simple and parameterized macros, conditionally include text, perform conditional evaluation, perform simple string operations, perform simple integer arithmetic and much more.

**Note:** GPFS uses the **m4** built-in **changequote** macro to change the quote pair to [ ] and the **changecom** macro to change the comment pair to /* */ (as in the C programming language).

Utilizing **m4** as a front-end processor simplifies writing policies and produces policies that are easier to understand and maintain. Here is Example 14 on page 325 from "Policy rules: Examples and tips" on page 323 written with a few **m4** style macro definitions:

```
define(access_age,(DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME)))

define(weight_expression,
       CASE
        WHEN access_age > 365
          THEN 100000 + access_age
        WHEN access_age < 30
          THEN 0
        ELSE
          KB_ALLOCATED
        END
)

RULE year-old MIGRATE FROM POOL pool_X
   THRESHOLD(90,50) WEIGHT(weight_expression)
   TO POOL pool_ZZ
   WHERE access_age > 365

RULE 3month-old MIGRATE FROM POOL pool_X
   THRESHOLD(90,80) WEIGHT(weight_expression)
   TO POOL pool_ZZ
   WHERE access_age > 90
```

If you would like to use megabytes or gigabytes instead of kilobytes to represent file sizes, and **SUNDAY**, **MONDAY**, and so forth instead of 1, 2, and so forth to represent the days of the week, you can use macros and rules like this:

```
define(MB_ALLOCATED,(KB_ALLOCATED/1024.0))
define(GB_ALLOCATED,(KB_ALLOCATED/1048576.0))
define(SATURDAY,7)
define(SUNDAY,1)
define(MONDAY,2)
define(DAY_OF_WEEK, DayOfWeek(CURRENT_DATE))

RULE 'gb1' WHEN(DAY_OF_WEEK IN (SATURDAY,SUNDAY))
        MIGRATE TO POOL 'ypooly' WHERE GB_ALLOCATED >= .015

RULE 'mb4' MIGRATE TO POOL 'zpoolz' WHERE MB_ALLOCATED >= 4
```

The **mmapplypolicy** command provides a **-M** option that can be used to specify **m4** macro definitions when the command is invoked. The policy rules may include variable identifiers whose values can be set using one or more **-M** options on the **mmapplypolicy** command. The policy rules could then compare file attributes to the currently provided values for the macro defined variables.

Among other things, this allows you to create a single policy file and reuse it for incremental backups without editing the file for each backup. For example, if your policy file contains the rules:

```
RULE EXTERNAL POOL 'archive' EXEC '/opts/hpss/archiveScript' OPTS '-server archive_server'
RULE 'mig1' MIGRATE TO POOL 'dead' WHERE ACCESS_TIME < TIMESTAMP(deadline)
RULE 'bak1' MIGRATE TO POOL 'archive' WHERE MODIFICATION_SNAPID > last_snapid
```

Then, if you invoke **mmapplypolicy** with these options

```
mmapplypolicy ... -M "deadline='2006-11-30'" -M "last_snapid=SNAPID('2006_DEC')" \
-M archive_server="archive.abc.com"
```

The "`mig1`" rule will migrate old files that were not accessed since 2006/11/30 to an online pool named "dead". The "`bak1`" rule will migrate files that have changed since the 2006_DEC snapshot to an external pool named "`archive`". When the external script `/opts/hpss/archiveScript` is invoked, its arguments will include "`-server archive.abc.com`".

## Managing policies

Policies and the rules that they contain are used to assign files to specific storage pools.

A storage pool typically contains a set of volumes that provide a specific quality of service for a specific use, such as to store all files for a particular application or a specific business division.

Managing policies includes:

- "Creating a policy"
- "Installing a policy" on page 328
- "Changing the active policy" on page 329
- "Listing policies" on page 329
- "Validating policies" on page 329
- "Deleting policies" on page 329

### Creating a policy

Create a text file for your policy by following these guidelines.

- A policy must contain at least one rule.
- A policy file is limited to a size of 1 MB.
- The last placement rule of a policy rule list should be of this form, so that if no other placement rules apply to a file, the file will be assigned to a default pool:

```
RULE 'DEFAULT' SET POOL 'default-data-pool'
```

If you do not do this, and no other rule applies, an **EINVAL** error code is returned.

**For file systems that are upgraded to 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.

**For file systems that are upgraded to V4.1.1:** Until a file system is upgraded, if no **SET POOL** rules are present (set by **mmchpolicy**) for the file system, all data is stored in the **'system'** pool.

- Comments within a policy must start with a **/*** and end with a ***/**:

```
/* This is a comment */
```

For more information, see the topic "Policy rules" on page 301.

## Installing a policy

Install a policy by following these guidelines.

To install a policy:

1. Create a text file containing the desired policy rules.
2. Issue the **mmchpolicy** command.

## Using thresholds to migrate data between pools

Exhausting space in any one online storage pool generates a NO_SPACE event even though there might be space available in other online storage pools. To create free space, file data can be moved to other online storage pools, deleted, or moved to external storage pools.

Tiered storage solutions can avoid **NO_SPACE** events by monitoring file system space usage and migrating data to other storage pools when the system exceeds a specified threshold. Policies can be used to monitor space usage using a threshold by using the **THRESHOLD** keyword in the policy along with a high water mark and a low water mark. A threshold policy can be applied to a file system by using the **mmchpolicy** command.

A NO_SPACE event is generated if the file system is out of space. A **lowDiskSpace** event requires a threshold. If a threshold is specified, a **lowDiskSpace** event is generated.

GPFS provides user exits for NO_SPACE and **lowDiskSpace** events. Using the **mmaddcallback** command, you can specify a script that runs when either of these events occurs. For more information, see the topic *mmaddcallback command* in the *IBM Spectrum Scale: Command and Programming Reference*.

The file with the policy rules used by **mmapplypolicy** is the one that is currently installed in the file system. It is a good idea for the HSM user to define migration or deletion rules to reduce the usage in each online storage pool. Migration rules that are defined with a high and low THRESHOLD establish the threshold that is used to signal the **lowDiskSpace** event for that pool. Because more than one migration rule can be defined, the threshold for a pool is the minimum of the high thresholds set by the rules for that pool. Each pool has its own threshold. Pools without migration rules do not signal a **lowDiskSpace** event.

In order to enable the low space events required for the policy to work, the **enableLowspaceEvents** global parameter must be set to 'yes'. To view the current status of this setting, run **mmlsconfig** and **mmchconfig enableLowspaceEvents=yes** to set it if necessary.

| **Note:** GPFS must be restarted on all nodes in order for this setting to take effect.
| A callback must be added in order to trigger the policy run when the low space event is generated. A
| simple way to add the callback is using the **mmstartpolicy** command.

| To add a callback, run this command. The following command is on one line:

```
| mmaddcallback MIGRATION --command /usr/lpp/mmfs/bin/mmstartpolicy --event lowDiskSpace
| --parms "%eventName %fsName --single-instance
```

| The **--single-instance** flag is required to avoid running multiple migrations on the file system at the same
| time.

## Changing the active policy

When you prepare a file with the new or changed policy rules, then issue the **mmchpolicy** command.

The **mmchpolicy** command activates the following sequence of events:

1. The policy file is read into memory, and the information is passed to the current file system manager node.
2. The policy rules are validated by the file system manager.
3. If the policy file contains incorrect rules, no updates are made and an error is returned.
4. If no errors are detected, the new policy rules are installed in an internal file.

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.

## Listing policies

When you use the **mmlspolicy** command to list policies, follow these guidelines.

The **mmlspolicy** command displays policy information for a given file system. The information displayed is:

- When the policy file was installed.
- The user who installed the policy file.
- The first line of the original policy file.

The **mmlspolicy -L** command returns the installed (original) policy file. This shows all the rules and comments as they were in the policy file when it was installed. This is useful if you want to change policy rules - simply retrieve the original policy file using the **mmlspolicy -L** command and edit it.

## Validating policies

When you validate a policy file, follow this guideline.

The **mmchpolicy -I test** command validates but does *not* install a policy file.

## Deleting policies

When you remove the current policy rules and restore the file-placement policy, follow this guideline.

To remove the current policy rules and restore the default GPFS file-placement policy, specify **DEFAULT** as the name of the policy file on the **mmchpolicy** command. This is equivalent to installing a policy file with just one rule:

```
RULE 'DEFAULT' SET POOL 'system'
```

## Improving performance with the --sort-command parameter

To improve performance of the **mmapplypolicy** command, follow these guidelines.

One possible way to improve the performance of the **mmapplypolicy** command is to specify an alternative sort command to be used instead of the default sort command provided by the operating system. To do this, issue **mmapplypolicy --sort-command** *SortCommand*, specifying the executable path of the alternative command.

For example, on AIX the GNU **sort** program, freely available within the `coreutils` package from AIX Toolbox for Linux Applications (www.ibm.com/systems/power/software/aix/linux/toolbox), will typically perform large sorting tasks much faster than the standard AIX sort command. If you wanted to specify the GNU sort program, you would use the following command: `mmapplypolicy --sort-command /opt/freeware/bin/sort`.

Before issuing **mmapplypolicy --sort-command** on a large number of files, first do a performance comparison between the alternative sort command and the default sort command on a smaller number of files to determine whether the alternative command is in fact faster.

If you specify an alternative sort command, it is recommended that you install it on all cluster nodes.

# Working with external storage pools

With external storage pools you can migrate files to storage pools managed by an external application such as IBM Spectrum Protect.

The following topics describe how to work with external storage pools:

- Defining the external pools
- "User-provided program for managing external pools" on page 331
- "File list format" on page 331
- "Record format" on page 332
- "Migrate and recall with external pools" on page 333
- "Pre-migrating files with external storage pools" on page 334
- "Purging files from external storage pools" on page 334
- "Using thresholds to migrate data between pools" on page 328

## Defining external pools

When you define external pools, follow these rules.

GPFS file management policy rules control data migration into external storage pools. Before you can write a migration policy you must define the external storage pool that the policy will reference. After you define the storage pool, you can then create policies that set thresholds that trigger data migration into or out of the referenced external pool.

When a storage pool reaches the defined threshold or when you invoke **mmapplypolicy**, GPFS processes the metadata, generates a list of files, and invokes a user provided script or program which initiates the appropriate commands for the external data management application to process the files. This allows GPFS to transparently control offline storage and provide a tiered storage solution that includes tape or other media.

Before you can migrate data to an external storage pool, you must define that pool. To define external storage pools, use a GPFS policy rule as follows:

```
RULE EXTERNAL POOL 'PoolName' EXEC 'InterfaceScript' [OPTS 'OptionsString'] [ESCAPE 'SpecialCharacters']
```

Where:

- *PoolName* defines the name of the storage pool
- *InterfaceScript* defines the program or script to be invoked to migrate data to or from the external pool
- *OptionsString* is an optional string that, if provided, will be passed to the *InterfaceScript*

You must have a separate EXTERNAL POOL rule for each external pool that you wish to define.

### Example of a rule that defines a storage pool

The following rule defines a storage pool called `externalpoolA`.

```
RULE EXTERNAL POOL 'externalpoolA' EXEC '/usr/hsm/bin/hsmControl' OPTS '-server=hsm-manager.nyc.com'
```

In this example:

- `externalpoolA` is the name of the external pool
- `/usr/hsm/bin/hsmControl` is the location of the executable script that will be invoked when there are files for migration
- `-server=hsm-manager.nyc.com` is the location of storage pool `externalpoolA`

For additional information, refer to "User-provided program for managing external pools."

## User-provided program for managing external pools

After you define an external storage pool, subsequent migration or deletion rules might refer to that pool as a source or target storage pool.

When the **mmapplypolicy** command is invoked and a rule dictates that data should be moved to or from an external pool, the user provided program identified with the **EXEC** clause in the policy rule launches. That executable program receives three arguments:

- The command to be executed. Your script should implement each of the following sub-commands:
  - LIST - Provides arbitrary lists of files with no semantics on the operation.
  - MIGRATE - Migrate files to external storage and reclaim the online space allocated to the file.
  - PREMIGRATE - Migrate files to external storage but do not reclaim the online space.
  - PURGE - Delete files from both the online file system and the external storage.
  - RECALL - Recall files from external storage to the online storage.
  - TEST – Test for presence and operation readiness. Return zero for success. Return non-zero if the script should not be used on a given node.
- The name of a file containing a list of files to be migrated, premigrated, or purged. See "File list format" for detailed description of the layout of the file.
- Any optional parameters specified with the OPTS clause in the rule. These optional parameters are not interpreted by the GPFS policy engine.

The **mmapplypolicy** command invokes the external pool script on all nodes in the cluster that have installed the script in its designated location. The script must be installed at the node that runs **mmapplypolicy**. You can also install the script at other nodes for parallel operation but that is not required. GPFS may call your exit script one or more times for each command.

**Important:** Use the **EXCLUDE** rule to exclude any special files that are created by an external application. For example, when using IBM Spectrum Protect or Hierarchical Storage Management (HSM), exclude the **.SpaceMan** directory to avoid migration of **.SpaceMan**, which is an HSM repository.

## File list format

Each call to the external pool script specifies the pathname for a temporary file that contains a list of files to be operated on.

This file list defines one file per line as follows:

```
InodeNumber GenNumber SnapId [OptionalShowArgs] -- FullPathToFile
```

where:

- *InodeNumber* is a 64-bit inode number.

- *GenNumber* is a 32-bit file generation number.
- *SnapId* is a 64-bit snapshot identifier.
- *OptionalShowArgs* is the result, if any, from the evaluation of the SHOW clause in the policy rule.
- *FullPathToFile* is a fully qualified path name to the file. When there are multiple paths within a file system to a particular file (*Inode*, *GenNumber*, and *SnapId*), each path is shown.
- The "--" characters are a field delimiter that separates the optional show parameters from the path name to the file.

**Note:** GPFS does not restrict the character set used for path and file names. All characters except '\0' are valid. To make the files readily parsable, files or directories containing the newline character and/or other special characters are “escaped”, as described previously, in connection with the `ESCAPE '%special-characters'` clause.

## Record format

The format of the records in each file list file can be expressed as shown in the following example.

Each file list file:

```
iAggregate:WEIGHT:INODE:GENERATION:SIZE:iRule:resourceID:attr_flags:
path-length!PATH_NAME:pool-length!POOL_NAME
[;show-length>!SHOW]end-of-record-character
```

where:

- *iAggregate* is a grouping index that is assigned by **mmapplypolicy**.
- *WEIGHT* represents the **WEIGHT** policy language file attribute.
- *INODE* represents the **INODE** policy language file attribute.
- *GENERATION* represents the **GENERATION** policy language file attribute.
- *SIZE* represents the **SIZE** policy language file attribute.
- *iRule* is a rule index number assigned by **mmapplypolicy**, which relates to the policy rules file that is supplied with the **-P** argument.
- *resourceID* represents a pool index, **USER_ID**, **GROUP_ID**, or fileset identifier, depending on whether thresholding is done with respect to pool usage or to user, group, or fileset quotas.
- *attr_flags* represents a hexadecimal encoding of some of the attributes that are also encoded by the policy language variable *MISC_ATTRIBUTES*. The low-order 20 bits of *attr_flags* are taken from the **ia_flags** word that is defined in the **gpfs.h** API definition.
- *path-length* represents the length of the character string PATH_NAME.
- *pool-length* represents the length of the character string POOL_NAME.
- *show-length* represents the length of the character string SHOW.
- *end-of-record-character* is **\n** or **\0**.

**Note:** You can only change the values of the *iAggregate*, *WEIGHT*, *SIZE*, and *attr_flags* fields. Changing the values of other fields can cause unpredictable policy execution results.

All of the numeric fields are represented as hexadecimal strings, except the *path-length*, *pool-length*, and *show-length* fields, which are decimal encoded. These fields can be preceded by a minus sign ( **-** ), which indicates that the string that follows it contains escape sequences. In this case, the string might contain occurrences of the character pair **\n**, which represents a single newline character with a hexadecimal value of 0xA in the filename. Also, the string might contain occurrences of the character pair \\, which represents a single \ character in the filename. A \ will only be represented by \\ if there are also newline characters in the filename. The value of the length field within the record counts any escape characters.

The encoding of *WEIGHT* is based on the 64-bit IEEE floating format, but its bits are *flipped* so that when a file list is sorted using a conventional collating sequence, the files appear in decreasing order, according to their *WEIGHT*.

The encoding of *WEIGHT* can be expressed and printed using C++ as:

```
double w =  - WEIGHT;
/* This code works correctly on big-endian and little-endian systems */
uint64 u = *(uint64*)&w;  /* u is a 64 bit long unsigned integer
     containing the IEEE 64 bit encoding of the double floating point
     value of variable w */
     uint64 hibit64 = ((uint64)1<<63);
if (w < 0.0) u = ~u;  /* flip all bits */
else u = u | hibit64; /* force the high bit from 0 to 1,
      also handles both "negatively" and "positively" signed 0.0 */
printf("%016llx",u);
```

The format of the majority of each record can be expressed in C++ as:

```
printf("%03x:%016llx:%016llx:%llx:%llx:%x:%x:%llx:%d!%s:%d!%s",
  iAggregate, u /*encoding of -1*WEIGHT from above*/, INODE, ... );
```

Notice that the first three fields are fixed in length to facilitate the sorting of the records by the field values *iAggregate*, *WEIGHT*, and *INODE*.

The format of the optional SHOW string portion of the record can be expressed as:

```
if(SHOW && SHOW[0]) printf(";%d!%s",strlen(SHOW),SHOW);
```

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

## Migrate and recall with external pools

After you define an external storage pool, subsequent migration or deletion rules might refer to that pool as a source or target storage pool.

When you invoke **mmapplypolicy** and a rule dictates that data should be deleted or moved to or from an external pool, the program identified in the EXTERNAL POOL rule is invoked with the following arguments:

- The command to be executed.
- The name of the file containing a list of files to be migrated, pre-migrated, or purged.
- Optional parameters, if any.

For example, let us assume an external pool definition:

```
RULE EXTERNAL POOL 'externalpoolA'
   EXEC '/usr/hsm/bin/hsmControl' OPTS '-server=hsm-manager.nyc.com'
```

To move files from the internal **system** pool to storage pool "externalpoolA" you would simply define a migration rule that may look something like this:

```
RULE 'MigToExt' MIGRATE FROM POOL('system') TO POOL('externalpoolA') WHERE ...
```

This would result in the external pool script being invoked as follows:

```
/usr/hsm/bin/hsmControl MIGRATE /tmp/filelist -server=hsm-manager.nyc.com
```

Similarly, a rule to migrate data from an external pool back to an internal storage pool could look like:

```
RULE 'MigFromExt' MIGRATE FROM POOL 'externalpoolA' TO POOL 'system' WHERE ...
```

This would result in the external pool script being invoked as follows:

```
/usr/hsm/bin/hsmControl RECALL /tmp/filelist -server=hsm-manager.nyc.com
```

**Notes:**

1. When migrating to an external storage pool, GPFS ignores the LIMIT and REPLICATION clauses in the policy rule.
2. If you are using HSM with external storage pools, you may need to create specific rules to avoid system problems. These rules should exclude HSM-related system files from both migration and deletion. These rules use the form:

```
RULE 'exclude hsm system files' EXCLUDE WHERE PATH_NAME LIKE '%/.SpaceMan%'
```

## Pre-migrating files with external storage pools

Pre-migration is a standard technique of Hierarchical Storage Management (HSM) systems such as IBM Spectrum Protect.

Pre-migration copies data from GPFS internal storage pools to external pools but leaves the original data online in the active file system. Pre-migrated files are often referred to as "dual resident" to indicate that the data for the files are available both online in GPFS and offline in the external storage manager. Files in the pre-migrated state allow the external storage manager to respond more quickly to low space conditions by simply deleting the copy of the file data that is stored online.

The files to be pre-migrated are determined by the policy rules that migrate data to an external storage pool. The rule will select files to be migrated and optionally select additional files to be pre-migrated. The THRESHOLD clause of the rule determines the files that need to be pre-migrated.

If you specify the THRESHOLD clause in file migration rules, the **mmapplypolicy** command selects files for migration when the affected storage pool reaches the specified high occupancy percentage threshold. Files are migrated until the storage pool utilization is reduced to the specified low occupancy percentage threshold. When migrating to an external storage pool, GPFS allows you to specify a third pool occupancy percentage which defines the file pre-migration threshold: after the low occupancy percentage is reached, files are pre-migrated until the pre-migration occupancy percentage is reached.

To explain thresholds in another way, think of an internal storage pool with a high threshold of 90%, a low threshold of 80%, and a pre-migrate threshold of 60%. When this internal storage pool reaches 90% occupancy, the policy rule will migrate files until the occupancy of the pool reaches 80% then it will continue to pre-migrate another 20% of the file space until the 60% threshold is reached.

Pre-migration can only be done with external storage managers using the XDSM Data Storage Management API (DMAPI). Files in the migrated and pre-migrated state will have a DMAPI managed region set on the file data. Files with a managed region are visible to **mmapplypolicy** and may be referenced by a policy rule. You can approximate the amount of pre-migrated space required by counting the space used after the end of the first full data block on all files with managed regions.

**Note:**

1. If you do not set a pre-migrate threshold or if you set a value that is greater than or equal to the low threshold, then GPFS will not pre-migrate files. This is the default setting.
2. If you set the pre-migrate threshold to zero, then GPFS will pre-migrate all files.

## Purging files from external storage pools

Files that have been migrated to an external storage pool continue to have their file name and attributes stored in GPFS; only the file data has been migrated. Files that have been migrated or pre-migrated to an external storage pool may be deleted from the GPFS internal storage pool and from the external storage pool with the policy language using a DELETE rule.

```
RULE 'DelFromExt' DELETE WHERE ...
```

If the file has been migrated or pre-migrated, this would result in the external pool script being invoked as follows:

```
/usr/hsm/bin/hsmControl PURGE /tmp/filelist -server=hsm-manager.nyc.com
```

The script should delete a file from both the online file system and the external storage manager. However, most HSM systems automatically delete a file from the external storage manager whenever the online file is deleted. If that is how your HSM system functions, your script will only have to delete the online file.

## Backup and restore with storage pools

When you back up data or restore data to a storage pool, consider the following descriptions.

You can use the GPFS ILM tools to backup data for disaster recovery or data archival to an external storage manager such as the IBM Spectrum Protect Backup-Archive client. When backing up data, the external storage manager must preserve the file name, attributes, extended attributes, and the file data. Among other things, the extended attributes of the file also contain information about the assigned storage pool for the file. When you restore the file, this information is used to assign the storage pool for the file data.

The file data may be restored to the storage pool to which it was assigned when it was backed up or it may be restored to a pool selected by a restore or placement rule using the backed up attributes for the file. GPFS supplies three subroutines that support backup and restore functions with external pools:

- **gpfs_fgetattrs()**
- **gpfs_fputattrs()**
- **gpfs_fputattrswithpathname()**

GPFS exports the extended attributes for a file, including its ACLs, using **gpfs_fgetattrs()**. Included in the extended attributes is the name of the storage pool to which the file has been assigned, as well as file attributes that are used for file placement. When the file is restored the extended attributes are restored using either **gpfs_fputattrs()** or **gpfs_fputattrswithpathname()**.

When a backup application uses **gpfs_fputattrs()** to restore the file, GPFS assigns the restored file to the storage pool with the same name as when the file was backed up. Thus by default, restored files are assigned to the same storage pool they were in when they were backed up. If that pool is not available, GPFS tries to select a pool using the current file placement rules. If that fails, GPFS assigns the file to the system storage pool.

**Note:** If a backup application uses **gpfs_fputattrs()** to restore a file, it will omit the **RESTORE** RULE.

When a backup application restores the file using **gpfs_fputattrswithpathname()**, GPFS is able to access additional file attributes that may have been used by placement or migration policy rules to select the storage pool for the file. This information includes the UID and GID for the owner, the access time for the file, file modification time, file size, the amount of storage allocated, and the full path to the file. GPFS uses **gpfs_fputattrswithpathname()** to match this information with restore policy rules you define.

In other words, the **RESTORE** rule looks at saved file attributes rather than the current file attributes. The call to **gpfs_fputattrswithpathname()** tries to match the saved information to a **RESTORE** rule. If the **RESTORE** rules cannot match saved attributes, GPFS tries to restore the file to the same storage pool it was in when the file was backed up. If that pool is not available GPFS tries to select a pool by matching placement rules. If that fails, GPFS assigns the file to the system storage pool.

**Note:** When a **RESTORE** rule is used, and restoring the file to the specified pool would exceed the occupancy percentage defined for that pool, GPFS skips that rule and the policy engine looks for the next rule that matches. While testing for matching rules, GPFS takes into account the specified replication factor and the **KB_ALLOCATED** attribute of the file that is being restored.

The **gpfs_fgetattrs()**, **gpfs_fputattrs()**, and **gpfs_fputattrswithpathname()** subroutines have optional flags that further control the selection of storage pools. For more information, see the topics *gpfs_fgetattrs() subroutine*, *gpfs_fputattrs() subroutine*, and *gpfs_fputattrswithpathname() subroutine* in the *IBM Spectrum Scale: Command and Programming Reference*.

### Working with external lists

External lists, like external pools, generate lists of files. For external pools, the operations on the files correspond to the rule that references the external pool. For external lists, there is no implied operation; it is simply a list of files that match the criteria specified in the policy rule.

External lists must be defined before they can be used. External lists are defined by:

```
RULE EXTERNAL LIST 'ListName' EXEC 'InterfaceScript' [OPTS 'OptionsString'] [ESCAPE 'SpecialCharacters']
```

Where:

- *ListName* defines the name of the external list
- *InterfaceScript* defines the program to be invoked to operate on the list of files
- *OptionsString* is an optional string that, if provided, will be passed to the *InterfaceScript*

See "User-provided program for managing external pools" on page 331.

#### Example

The following rule defines an external list called `listfiles`:

```
RULE EXTERNAL LIST 'listfiles' EXEC '/var/mmfs/etc/listControl' OPTS '-verbose'
```

In this example:

- `listfiles` is the name of the external list
- `/var/mmfs/etc/listControl` is the location of the executable script that defines the operations on the list of files
- `-verbose` is an optional flag to the `listControl` script

The EXTERNAL LIST rule provides the binding between the lists generated with regular LIST rules and the external program that you want to run with these lists as input. For example, this rule would generate a list of all files that have more than 1 MB of data in an internal storage pool:

```
RULE 'ListLargeFiles' LIST 'listfiles' WHERE KB_ALLOCATED > 1024
```

By default, only user files are included in lists. To include directories, symbolic links, and other file system objects, the DIRECTORIES_PLUS clause must be specified. For example, this rule would generate a list of all objects in the file system.

```
RULE 'ListAllObjects' LIST 'listfiles' DIRECTORIES_PLUS
```

## Filesets

In most file systems, a file hierarchy is represented as a series of directories that form a tree-like structure. Each directory contains other directories, files, or other file-system objects such as symbolic links and hard links. Every file system object has a name associated with it, and is represented in the namespace as a node of the tree.

In addition, GPFS utilizes a file system object called a *fileset*. A fileset is a subtree of a file system namespace that in many respects behaves like an independent file system. Filesets provide a means of partitioning the file system to allow administrative operations at a finer granularity than the entire file system:

- Filesets can be used to define quotas on both data blocks and inodes.

- The owning fileset is an attribute of each file and can be specified in a policy to control initial data placement, migration, and replication of the file's data. See “Policies for automating file management” on page 300.
- Fileset snapshots can be created instead of creating a snapshot of an entire file system.

GPFS supports independent and dependent filesets. An independent fileset is a fileset with its own inode space. An inode space is a collection of inode number ranges reserved for an independent fileset. An inode space enables more efficient per-fileset functions, such as fileset snapshots. A dependent fileset shares the inode space of an existing, independent fileset. Files created in a dependent fileset are assigned inodes in the same collection of inode number ranges that were reserved for the independent fileset from which it was created.

When the file system is created, only one fileset, called the *root* fileset, exists. The root fileset is an independent fileset that cannot be deleted. It contains the root directory as well as any system files such as quota files. As new files and directories are created, they automatically become part of the parent directory's fileset. The fileset to which a file belongs is largely transparent for ordinary file access, but the containing fileset can be displayed along with the other attributes of each file using the **mmlsattr -L** command.

The root directory of a GPFS file system is also the root of the root fileset.

## Fileset namespace

A newly created fileset consists of an empty directory for the root of the fileset, and it is initially not linked into the file system's namespace. A newly created fileset is not visible to the user until it is attached to the namespace by issuing the **mmlinkfileset** command.

Filesets are attached to the namespace with a special link called a *junction*. A junction is a special directory entry, much like a POSIX hard link, that connects a name in a directory of one fileset (source) to the root directory of another fileset (target). A fileset may be the target of only one junction, so that a fileset has a unique position in the namespace and a unique path to any of its directories. The target of the junction is referred to as the *child fileset*, and a fileset can have any number of children. 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.

Once a fileset has been created and linked into the namespace, an administrator can unlink the fileset from the namespace by issuing the **mmunlinkfileset** command. This makes all files and directories within the fileset inaccessible. If other filesets were linked below it, the other filesets become inaccessible, but they do remain linked and will become accessible again when the fileset is re-linked. Unlinking a fileset, like unmounting a file system, fails if there are open files. The **mmunlinkfileset** command has a force option to close the files and force the unlink. If there are open files in a fileset and the fileset is unlinked with the force option, future references to those files will result in **ESTALE** errors. Once a fileset is unlinked, it can be re-linked into the namespace at its original location or any other location (it cannot be linked into its children since they are not part of the namespace while the parent fileset is unlinked).

The namespace inside a fileset is restricted to a single, connected subtree. In other words, a fileset has only one root directory and no other entry points such as hard links from directories in other filesets. Filesets are always connected at the root directory and only the junction makes this connection. Consequently, hard links cannot cross fileset boundaries. Symbolic links, of course, can be used to provide shortcuts to any file system object in the namespace.

The root fileset is an exception. The root fileset is attached to the local namespace using the standard **mount** command. It cannot be created, linked, unlinked or deleted using the GPFS fileset commands.

See “Managing filesets” on page 340.

## Filesets and quotas

The GPFS quota commands support the **-j** option for fileset block and inode allocation.

The quota limit on blocks and inodes in a fileset are independent of the limits for specific users or groups of users. See the following command in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmdefedquota**
- **mmdefedquotaon**
- **mmdefedquotaoff**
- **mmedquota**
- **mmlsquota**
- **mmquotaoff**
- **mmquotaon**
- **mmrepquota**

In addition, see the description of the **--perfileset-quota** parameter of the following commands:

- **mmchfs**
- **mmcrfs**
- **mmlsfs**

## Filesets and storage pools

Filesets are not specifically related to storage pools, although each file in a fileset physically resides in blocks in a storage pool. This relationship is many-to-many; each file in the fileset can be stored in a different user storage pool.

A storage pool can contain files from many filesets. However, all of the data for a particular file is wholly contained within one storage pool.

Using file-placement policies, you can specify that all files created in a particular fileset are to be stored in a specific storage pool. Using file-management policies, you can define how files in a specific fileset are to be moved or deleted during the file's life cycle. See “Policy rules: Terms” on page 303.

## Filesets and global snapshots

A GPFS global snapshot preserves the contents of the entire file system, including all its filesets, even unlinked ones.

The state of filesets in the snapshot is unaffected by changes made to filesets in the active file system, such as unlink, link or delete. The saved file system can be accessed through the **.snapshots** directories and the namespace, including all linked filesets, appears as it did when the snapshot was created. Unlinked filesets are inaccessible in the snapshot, as they were in the active file system. However, restoring a snapshot also restores the unlinked filesets, which can then be re-linked and accessed.

If a fileset is included in a global snapshot, it can be deleted but it is not entirely removed from the file system. In this case, the fileset is emptied of all contents and given a status of 'deleted'. The contents of a fileset remain available in the snapshots that include the fileset (that is, through some path containing a **.snapshots** component) even after the fileset is deleted, since all the contents of the fileset are saved when a snapshot is created. The fileset remains in the deleted state until the last snapshot containing it is deleted, at which time the fileset is automatically deleted.

A fileset is included in a global snapshot if the snapshot is created after the fileset was created. Deleted filesets appear in the output of the **mmlsfileset** and **mmlsfileset --deleted** commands, and the **-L** option can be used to display the latest snapshot that includes a fileset.

During a restore from a global snapshot, attributes of filesets included in the snapshot can be altered. The filesets included in the global snapshot are restored to their former state, and newer filesets are deleted. Also, restore may undelete deleted filesets and change linked filesets to unlinked or vice versa. If the name of a fileset was changed since the snapshot was taken, the old fileset name will be restored.

## Fileset-level snapshots

Instead of creating a global snapshot of an entire file system, a fileset snapshot can be created to preserve the contents of a single independent fileset plus all dependent filesets that share the same inode space.

If an independent fileset has dependent filesets that share its inode space, then a snapshot of the independent fileset will also include those dependent filesets. In other words, a fileset snapshot is a snapshot of the whole inode space.

Each independent fileset has its own hidden **.snapshots** directory in the root directory of the fileset that contains any fileset snapshots. The **mmsnapdir** command allows setting an option that makes global snapshots also available through **.snapshots** in the root directory of all independent filesets. The **.snapshots** directory in the file system root directory lists both global snapshots and fileset snapshots of the root fileset (the root fileset is an independent fileset). This behavior can be customized with the **mmsnapdir** command.

Fileset snapshot names need not be unique across different filesets, so it is valid to use the same name for fileset snapshots of two different filesets because they will appear under **.snapshots** in two different fileset root directories.

You can restore independent fileset snapshot data and attribute files with the **mmrestorefs** command. For complete usage information, see the topic *mmrestorefs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Filesets and backup

The **mmbackup** command and IBM Spectrum Protect are unaware of the existence of filesets. When restoring a file system that had been backed up to IBM Spectrum Protect, the files are restored to their original path names, regardless of the filesets of which they were originally a part.

IBM Spectrum Protect has no mechanism to create or link filesets during restore. Therefore, if a file system is migrated to IBM Spectrum Protect and then filesets are unlinked or deleted, restore or recall of the file system does not restore the filesets.

During a full restore from backup, all fileset information is lost and all files are restored into the root fileset. It is recommended that you save the output of the **mmlsfileset** command to aid in the reconstruction of fileset names and junction locations. Saving **mmlsfileset -L** also allows reconstruction of fileset comments. Both command outputs are needed to fully restore the fileset configuration.

A partial restore can also lead to confusion if filesets have been deleted, unlinked, or their junctions moved, since the backup was made. For example, if the backed up data was in a fileset that has since been unlinked, the restore process puts it into files and directories in the parent fileset. The unlinked fileset cannot be re-linked into the same location until the restored data is moved out of the way. Similarly, if the fileset was deleted, restoring its contents does not recreate the deleted fileset, but the contents are instead restored into the parent fileset.

Since the **mmbackup** command operates by traversing the directory structure, it does not include the contents of unlinked filesets, even though they are part of the file system. If it is desired to include these filesets in the backup, they should be re-linked, perhaps into a temporary location. Conversely, temporarily unlinking a fileset is a convenient mechanism to exclude it from a backup.

**Note:** It is recommended not to unlink filesets when doing backups. Unlinking a fileset during an **mmbackup** run can cause the following:

- failure to back up changes in files that belong to an unlinked fileset
- expiration of files that were backed up in a previous **mmbackup** run

In summary, fileset information should be saved by periodically recording **mmlsfileset** output somewhere in the file system, where it is preserved as part of the backup process. During restore, care should be exercised when changes in the fileset structure have occurred since the backup was created.

**Attention:** If you are using the IBM Spectrum Protect Backup-Archive client you must 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 TSM server which may result in the loss of backup data during the expiration process.

# Managing filesets

Managing your filesets includes:



## Creating a fileset

Filesets are created with the **mmcrfileset** command.

By default, filesets are created as dependent filesets that share the root's inode space. The **--inode-space** *ExistingFileset* option can be used to create a dependent fileset that will share inode space with an existing fileset. The *ExistingFileset* specified can be **root** or any other independent fileset, but cannot be the reserved keyword **new**. The **--inode-space new** option can be used to create an independent fileset with its own dedicated inode space.

A newly created fileset consists of an empty directory for the root of the fileset and it is initially not linked into the existing namespace. Consequently, a new fileset is not visible, nor can files be added to it, but the fileset name is valid and the administrator can establish quotas on it, or policies for it. The administrator must link the fileset into its desired location in the file system's namespace by issuing the **mmlinkfileset** command in order to make use of it.

After the fileset is linked, the administrator can change the ownership and permissions for the new fileset's root directory, which default to **root** and 0700, to allow users access to it. Files and directories copied into, or created within, the fileset's directory will become part of the new fileset.

Fileset names must follow these conventions:

- Are character strings and must be less than 256 characters in length.
- Must be unique within a file system.
- The name **root** is reserved for the fileset of the files system's root directory.

For complete usage information, see the topics *mmcrfileset command* and *mmlinkfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting a fileset

Filesets are deleted with the **mmdelfileset** command.

There are several notes to keep in mind when deleting filesets:

- 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 complete usage information, see the topics *mmdelfileset command*, *mmlsfileset command*, and *mmunlinkfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Linking a fileset

After the fileset is created, a junction must be created to link it to the desired location in the file system's namespace using the **mmlinkfileset** command.

The file system must be mounted in order to link a fileset. An independent fileset can be linked into only one location anywhere in the namespace, specified by the *JunctionPath* parameter:

- The root directory
- Any subdirectory
- The root fileset or to any other fileset

A dependent fileset can only be linked inside its own inode space.

If *JunctionPath* is not specified, the junction is created in the current directory and has the same name as the fileset being linked. After the command completes, the new junction appears as an ordinary directory, except that the user is not allowed to unlink or delete it with the **rmdir** command it. The user can 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 complete usage information, see the topic *mmlinkfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Unlinking a fileset

A junction to a fileset is removed with the **mmunlinkfileset** command, which unlinks the fileset only from the active directory namespace. The linked or unlinked state of a fileset in a snapshot is unaffected. The unlink fails if there are files open in the fileset, unless the **-f** option is specified. The root fileset cannot be unlinked.

After issuing the **mmunlinkfileset** command, the fileset can be re-linked to a different parent using the **mmlinkfileset** command. Until the fileset is re-linked, it is not accessible.

**Note:** If run against a file system that has an unlinked fileset, **mmapplypolicy** will not traverse the unlinked fileset.

**Attention:** If you are using the IBM Spectrum Protect Backup-Archive client you must 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 complete usage information, see the topic *mmunlinkfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

### Changing fileset attributes

To change a fileset's junction, you have to first unlink the fileset using the **mmunlinkfileset** command, and then create the new junction using the **mmlinkfileset** command.

To change the attributes of an existing fileset, including the fileset name, use the **mmchfileset** command.

**Note:** In an HSM-managed file system, moving or renaming migrated files between filesets will result in recalling of the date from the IBM Spectrum Protect server.

For complete usage information, see the topics *mmchfileset command*, *mmlinkfileset command*, and *mmunlinkfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

### Displaying fileset information

Fileset status and attributes are displayed with the **mmlsfileset** command.

Some of the attributes displayed include:

- Name of the fileset.
- Fileset identifier of the fileset.
- Junction path to the fileset.
- Status of the fileset.
- Root inode number of the fileset.
- Path to the fileset (if linked).
- Inode space.
- User provided comments (if any).

For complete usage information, see the topic *mmlsfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

To display the name of the fileset that includes a given file, run the **mmlsattr** command and specify the **-L** option. For complete usage information, see the topic *mmlsattr command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Immutability and appendOnly features

To prevent files from being changed or deleted unexpectedly, GPFS provides immutability and appendOnly restrictions.

## Applying immutability and appendOnly restrictions to individual files or to directories

You can apply immutability and appendOnly restrictions either to individual files within a fileset or to a directory.

An immutable file cannot be changed or renamed. An appendOnly file allows append operations, but not delete, modify, or rename operations.

An immutable directory cannot be deleted or renamed, and files cannot be added or deleted under such a directory. An appendOnly directory allows new files or subdirectories to be created with 0 byte length; all such new created files and subdirectories are marked as appendOnly automatically.

The **immutable** flag and the **appendOnly** flag can be set independently. If both immutability and appendOnly are set on a file, immutability restrictions will be in effect.

To set or unset these attributes, use the following command options:

**mmchattr -i yes|no**
: Sets or unsets a file to or from an immutable state.

  **-i yes** Sets the **immutable** attribute of the file to **yes**.

  **-i no** Sets the **immutable** attribute of the file to **no**.

**mmchattr -a yes|no**
: Sets or unsets a file to or from an appendOnly state.

  **-a yes** Sets the **appendOnly** attribute of the file to **yes**.

  **-a no** Sets the **appendOnly** attribute of the file to **no**.

**Note:** Before an immutable or appendOnly file can be deleted, you must change it to mutable or set appendOnly to **no** (by using the **mmchattr** command).

Storage pool assignment of an immutable or appendOnly file can be changed; an immutable or appendOnly file is allowed to transfer from one storage pool to another.

To display whether or not a file is immutable or appendOnly, issue this command:

```
mmlsattr -L myfile
```

The system displays information similar to the following:

```
file name:            myfile
metadata replication: 2 max 2
data replication:     1 max 2
immutable:            no
appendOnly:           no
flags:
storage pool name:    sp1
fileset name:         root
snapshot name:
creation Time:        Wed Feb 22 15:16:29 2012
Misc attributes:      ARCHIVE
```

## The effects of file operations on immutable and appendOnly files

Once a file has been set as immutable or appendOnly, the following file operations and attributes work differently from the way they work on regular files:

**delete** An immutable or appendOnly file cannot be deleted.

**modify/append**
An immutable file cannot be modified or appended

An appendOnly file cannot be modified, but it can be appended..

**Note:** The immutable and appendOnly flag check takes effect after the file is closed; therefore, the file can be modified if it is opened before the file is changed to immutable.

**mode** An immutable or appendOnly file's mode cannot be changed.

**ownership, acl**
These attributes cannot be changed for an immutable or appendOnly file.

**extended attributes**
These attributes cannot be added, deleted, or modified for an immutable or appendOnly file.

**timestamp**
The timestamp of an immutable or appendOnly file can be changed.

**directory**
If a directory is marked as immutable, no files can be created, renamed, or deleted under that directory. However, a subdirectory under an immutable directory remains mutable unless it is explicitly changed by **mmchattr**.

If a directory is marked as appendOnly, no files can be renamed or deleted under that directory. However, 0 byte length files can be created.

The following table shows the effects of file operations on an immutable file or an appendOnly file:

*Table 35. The effects of file operations on an immutable file or an appendOnly file*

| Operation | immutable | appendOnly |
|---|---|---|
| **Add**, **delete**, **modify**, or **rename** | No | No |
| **Append** | No | Yes |
| **Change ownership**, **mode**, or **acl** | No | No |
| **Change atime**, **mtime**, or **ctime** | Yes | Yes |
| **Add**, **delete**, or **modify extended attributes** | Disallowed by external methods such as **setfattr**.<br><br>Allowed internally for **dmapi**, **directio**, and others. | Same as for immutable. |
| **Create** a file under an immutable or appendOnly directory | No | Yes, 0 byte length only |
| **Rename** or **delete** a file under an immutable or appendOnly directory | No | No |
| **Modify** a mutable file under an immutable directory | Yes | Not applicable |
| Set an immutable file back to mutable | Yes | Not applicable |
| Set an appendOnly file back to a non-appendOnly state | Not applicable | Yes |

## Fileset-level integrated archive manager (IAM) modes

Fileset-level integrated archive manager (IAM) modes can be used to modify some of the file-operation restrictions that normally apply to immutable files (see Table 36 on page 345. The **mmchfileset**

**--iam-mode** parameter is used to set the IAM mode; for more information, see the topic *mmchfileset command* in the *IBM Spectrum Scale: Command and Programming Reference*.

*Table 36. IAM modes and their effect on immutable file operations*

| File Operation | Regular Mode | Advisory Mode | Noncompliant Mode | Compliant Mode |
|---|---|---|---|---|
| **Modify** | No | No | No | No |
| **Append** | No | No | No | No |
| **Rename** | No | No | No | No |
| Change ownership, acl | No | No | No | No |
| Change mode | No | No | No | No |
| Change **atime**, **mtime**, **ctime** | Yes | **mtime** and **ctime** can be changed.<br><br>**atime** is overloaded by expiration time.<br><br>Expiration time can be changed by using the **mmchattr --expiration-time** command (alternatively **mmchattr -E**) or touch. You can see the expiration time by using stat as **atime**. | Same as advisory mode | Same as advisory mode |
| Add, delete, or modify extended attributes. | Not allowed for external methods such as **setfattr**. Allowed internally for dmapi, directio, and etc. | Yes | Yes | Yes |
| Create, rename, or delete under an immutable directory | No | No | No | No |
| Modify mutable files under an immutable directory. | Yes | Yes | Yes | Yes |
| Retention rule enforced | No retention rule, cannot delete immutable files | No | Yes | Yes |
| Set ExpirationTime backwards | Yes | Yes | Yes | No |
| Delete an immutable file | No | Yes, always | Yes, only when expired | Yes, only when expired |
| Set an immutable file back to mutable | Yes | No | No | No |
| Allow hardlink | No for immutable or appendOnly files.<br><br>Yes for other files. | No | No | No |

*Table 36. IAM modes and their effect on immutable file operations (continued)*

| File Operation | Regular Mode | Advisory Mode | Noncompliant Mode | Compliant Mode |
|---|---|---|---|---|
| Rename or delete a non-empty directory | Yes for rename.<br><br>No for delete. | No for rename.<br><br>Yes for delete. | No for rename.<br><br>Yes for delete only if the immutable file has expired. | No for rename.<br><br>Yes for delete only if the immutable file has expired. |
| Remove user write permission to change a file to immutable | No | Yes | Yes | Yes |
| Display expiration time instead of **atime** for stat call | No | Yes | Yes | Yes |
| Set a directory to be immutable | Yes | No | No | No |

# Chapter 23. Creating and maintaining snapshots of file systems

A snapshot of an entire GPFS file system can be created to preserve the contents of the file system at a single point in time. Snapshots of the entire file system are also known as global snapshots. The storage overhead for maintaining a snapshot is keeping a copy of data blocks that would otherwise be changed or deleted after the time of the snapshot.

Snapshots of a file system are read-only; changes can only be made to the active (that is, normal, non-snapshot) files and directories.

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 that the snapshot 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.

**Notes:**

1. Because snapshots are not 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 topic *Recoverability considerations* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
2. Fileset snapshots provide a method to create a snapshot of an independent fileset instead of the entire file system. For more information about fileset snapshots, see "Fileset-level snapshots" on page 339.
3. A snapshot of a file creates a new file that captures the user data and user attributes from the original. The snapshot file is independent from the original file. For DMAPI managed file systems, the snapshot of a file is not automatically managed by DMAPI, regardless of the state of the original file. The DMAPI attributes from the original file are not inherited by the snapshot. For more information about DMAPI restrictions for GPFS, see the *IBM Spectrum Scale: Command and Programming Reference*.
4. When snapshots are present, deleting files from the active file system does not always result in any space actually being freed up; rather, blocks may be pushed to the previous snapshot. In this situation, the way to free up space is to delete the oldest snapshot. Before creating new snapshots, it is good practice to ensure that the file system is not close to being full.
5. The use of clones functionally provides writable snapshots. See Chapter 24, "Creating and managing file clones," on page 355.

Management of snapshots of a GPFS file system includes:



## Creating a snapshot

Use the **mmcrsnapshot** command to create a snapshot of an entire GPFS file system at a single point in time. Snapshots appear in the file system tree as hidden subdirectories of the root.

Global snapshots appear in a subdirectory in the root directory of the file system, whose default name is **.snapshots**. If you prefer to access 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. See "Linking to a snapshot" on page 350 for more information.

A snapshot of the file system, *Device*, is identified by a *SnapshotName* name on the **mmcrsnapshot** command. For example, given the file system **fs1** to create a snapshot **snap1**, enter:

```
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 only made 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
```

For complete usage information, see the topic *mmcrsnapshot command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Listing snapshots

Use the **mmlssnapshot** command to display existing snapshots of a file system and their attributes.

The **-d** option displays the amount of storage used by a snapshot. GPFS quota management does not take the data blocks used to store snapshots into account when reporting on and determining if quota limits have been exceeded. This is a slow operation and its usage is suggested for problem determination only.

For example, to display the snapshot information for the file system **fs1** with additional storage information, issue this command:

```
mmlssnapshot fs1 -d
```

The system displays information similar to:

```
Snapshots in file system fs1: [data and metadata in KB]
Directory               SnapId    Status          Created                        Data  Metadata
snap1                   1         Valid           Fri Oct 17 10:56:22 2003          0       512
```

For complete usage information, see the topic *mmlssnapshot command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Restoring a file system from a snapshot

Use the **mmrestorefs** command to restore user data and attribute files in an active file system from a snapshot.

Prior to issuing the **mmrestorefs** command, ensure that the file system is mounted. When restoring from an independent fileset snapshot, ensure that the fileset is in linked state.

Existing snapshots, including the one being used in the restore, are not modified by the **mmrestorefs** command. To obtain a snapshot of the restored file system, you must issue the **mmcrsnapshot** command to capture it before issuing the **mmrestorefs** command again.

As an example, suppose that you have a directory structure 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
```

If the directory **userA** is then deleted, the structure becomes similar to this:

```
/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 resulting 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
```

For complete usage information, see the topic *mmrestorefs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Reading a snapshot with the policy engine

You can use the policy engine to read the contents of a snapshot for backup purposes. The **mmapplypolicy** command provides the **-S** option to specify the snapshot during a policy run. Instead of matching rules to the active file system, the policy engine matches the rules against files in the snapshot.

**Notes:**

1. Snapshots are read-only. Policy rules such as MIGRATE or DELETE that make changes or delete files cannot be used with a snapshot.
2. An instance of **mmapplypolicy** can only scan one snapshot. Directing it at the `.snapshots` directory itself will result in a failure.

For complete usage information, see the topic *mmapplypolicy command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Linking to a snapshot

Snapshot root directories appear in a special **.snapshots** directory under the file system root.

If you prefer to link directly to the snapshot rather than always traverse the root directory, you can use the **mmsnapdir** command with the **-a** option to add a **.snapshots** subdirectory to all directories in the file system. These **.snapshots** subdirectories will contain a link into the corresponding directory for each snapshot that includes the directory in the active file system.

Unlike **.snapshots** in the root directory, however, the **.snapshots** directories added by the **-a** option of the **mmsnapdir** command are invisible in the sense that the **ls** command or **readdir()** function does not return **.snapshots**. This is to prevent recursive file system utilities such as **find** or **tar** from entering into the snapshot tree for each directory they process. For example, if you enter **ls -a /fs1/userA**, the **.snapshots** directory is not listed. However, you can enter **ls /fs1/userA/.snapshots** or **cd /fs1/userA/.snapshots** to confirm that **.snapshots** is present. If a user wants to make one of their snapshot directories more visible, it is suggested to create a symbolic link to **.snapshots**.

The inode numbers that are used for and within these special **.snapshots** directories are constructed dynamically and do not follow the standard rules. These inode numbers are visible to applications

through standard commands, such as **stat**, **readdir**, or **ls**. The inode numbers reported for these directories can also be reported differently on different operating systems. Applications should not expect consistent numbering for such inodes.

Specifying the **-r** option on the **mmsnapdir** command reverses the effect of the **-a** option, and reverts to the default behavior of a single **.snapshots** directory in the root directory.

The **-s** option allows you to change the name of the **.snapshots** directory. For complete usage information, see the topic *mmsnapdir command* in the *IBM Spectrum Scale: Command and Programming Reference*.

To illustrate this point, assume that a GPFS file system called **fs1**, which is mounted at **/fs1**, has one snapshot called **snap1**. The file system might appear similar to this:

```
/fs1/userA/file2b
/fs1/userA/file3b
/fs1/.snapshots/snap1/userA/file2b
/fs1/.snapshots/snap1/userA/file3b
```

To create links to the snapshots from each directory, and instead of **.snapshots**, use the name **.links**, enter:

```
mmsnapdir fs1 -a -s .links
```

After the command completes, the directory structure would appear similar to:

```
/fs1/userA/file2b
/fs1/userA/file3b
/fs1/userA/.links/snap1/file2b
/fs1/userA/.links/snap1/file3b

/fs1/.links/snap1/userA/file2b
/fs1/.links/snap1/userA/file3b
```

To delete the links, issue:

```
mmsnapdir fs1 -r
```

After the command completes, the directory structure is similar to the following:

```
/fs1/userA/file2b
/fs1/userA/file3b

/fs1/.links/snap1/userA/file2b
/fs1/.links/snap1/userA/file3b
```

For complete usage information, see the topic *mmsnapdir command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting a snapshot

Use the **mmdelsnapshot** command to delete GPFS snapshots of a file system.

For example, to delete **snap1** for the file system **fs1**, enter:

```
mmdelsnapshot fs1 snap1
```

The output is similar to this:

```
Invalidating snapshot files...
Deleting snapshot files...
 100.00 % complete on Tue Feb 28 10:40:59 2012
Delete snapshot snap1 complete, err = 0
```

For complete usage information, see the topic *mmdelsnapshot command* in the *IBM Spectrum Scale: Command and Programming Reference*.

# Managing snapshots with IBM Spectrum Scale GUI

Use **Files** > **Snapshots** page in the IBM Spectrum Scale GUI to manage snapshots through GUI.

Snapshots can be used in environments where multiple recovery points are necessary. A snapshot can be taken of file system or fileset data and then the data can be recovered from the snapshot if the production data becomes unavailable.

**Note:**

- Snapshots are read-only; changes can be made only to the normal and active files and directories, not to the snapshot.
- When a snapshot of an independent fileset is taken, only nested dependent filesets are included in the snapshot.

## Scheduling snapshot creation by using snapshot rules

You can either manually create the snapshots or create snapshot rules to automate the snapshot creation and retention.

To manually create a snapshot, click **Create Snapshot** in the Snapshots page and enter the required details under the **Manual** tab of the Create Snapshot window. Click **Create** after entering the details.

By creating a snapshot rule, you can automate the snapshot creation and retention. That is, in a snapshot rule you can specify a frequency in which the snapshots must be created and the number of snapshots that must be retained for a period. Snapshots are deleted only once in a day due to performance issues and the system determines which snapshots to be retained based on the retention policy. The retention policy helps to avoid unwanted storage of snapshots that result in waste of storage resources.

Retention policy has the following parameters:

- Frequency of snapshot creation
- Number of most recent snapshots to be retained. The most recent snapshot is identified based on the frequency of snapshot creation.
- Number of days for which you need to keep the latest snapshot of each day.
- Number of weeks for which you need to keep the latest snapshot of each week.
- Number of months for which you need to keep the latest snapshot of each month.

**Example scenario for retention policy**

The following table provides an example for the values that are specified against these parameters.

*Table 37. Example for retention period*

| Snapshot deletion time | Frequency | Minute | Number of most recent snapshots | Keep latest snapshots for | | | |
|---|---|---|---|---|---|---|---|
| | | | | Hours | Days | Weeks | Months |
| 2:30 AM | Hourly | 1 | 2 | 2 | 6 | 2 | 3 |

Based on the this retention rule, the following snapshots are created and retained on March 20, 2016 at 06:10 AM:

*Table 38. Example - Time stamp of snapshots that are retained based on the retention policy*

| Time stamp | Condition based on which snapshot is retained |
|---|---|
| December 31 (Thursday, 23:01 AM) | Keep latest snapshot for last 3 months |

*Table 38. Example - Time stamp of snapshots that are retained based on the retention policy (continued)*

| Time stamp | Condition based on which snapshot is retained |
|---|---|
| January 31 (Sunday, 23:01 AM) | Keep latest snapshot for last 3 months |
| February 29 (Monday, 23:01 AM) | Keep latest snapshot for last 3 months |
| March 5 (Saturday, 23:01 AM) | Keep latest snapshot for last 2 weeks |
| March 12 (Saturday, 23:01 AM) | Keep latest snapshot for last 2 weeks |
| March 14 (Monday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 15 (Tuesday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 16 (Wednesday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 17 (Thursday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 18 (Friday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 19 (Saturday, 23:01 AM) | Keep latest snapshot for last 6 days |
| March 20 (Sunday, 1:01 AM) | Keep 2 most recent |
| March 20 (Sunday, 2:01 AM) | Keep 2 most recent |
| March 20 (Sunday, 3:01 AM) | Created and retained because of the value that is selected for frequency is “Hourly” and the snapshot deletion time is set as 2:30. |
| March 20 (Sunday, 4:01 AM) | Created and retained because of the value that is selected for frequency is “Hourly” and the snapshot deletion time is set as 2:30. |
| March 20 (Sunday, 5:01 AM) | Created and retained because of the value that is selected for frequency is “Hourly” and the snapshot deletion time is set as 2:30. |
| March 20 (Sunday, 6:01 AM) | Created and retained because of the value that is selected for frequency is “Hourly” and the snapshot deletion time is set as 2:30. |

According to this rule, 17 snapshots are retained on March 20, 2016 at 06:10 AM.

To schedule snapshot creation and retention, perform the following steps:

1. Go to **Files > Snapshots**.
2. Click **Create Snapshot**.
3. In the Create Snapshot window, enter the path of the file system or independent fileset for which you need to create snapshots.
4. In the **Snapshot name** field, specify the name of the snapshot.
5. Click **Snapshot Rules**.
6. Click **Create Rule** to schedule the snapshot creation and retention. Create Snapshot Rule window is displayed.
7. In the **Name** field, type the name of the snapshot scheduling rule.
8. In the **Frequency** field, select the frequency in which you need to create snapshot. You need to enter some more details based on the value that is selected in the **Frequency** field. For example, if value selected is **Multiple Times an Hour**, select the minutes of the hour in which you need to create snapshots.
9. In the **Retention** fields, specify the number of snapshots that must be retained in a time period.
10. In the **Prefix** field, specify a prefix to be added with the name of the snapshots that are created with this rule.
11. Click **OK** to save the changes.

If you do not specify a name for the snapshot, the default name is given. The default snapshot ID is generated at the creation time by using the format "*@GMT-yyyy.MM.dd-HH.mm.ss*". If this option is given and the "*@GMT-date-time*" format is omitted, then this snapshot will not be identifiable by Windows VSS and the file restore is not possible by that method. Avoid white spaces, double and single quotation marks, the parentheses (), the star *, forward slash /, and backward slash \.

## Deleting snapshots

Use caution when removing snapshots that are associated with backups or when removing any file system snapshots that are used by any other component, such as Network Data Management Protocol (NDMP) or asynchronous replication.

For example, when you remove an NDMP snapshot, an NDMP backup that is defined to use the removed NDMP snapshot fails. If you remove an NDMP snapshot and the tracking file still exists, the NDMP backup attempts to use the removed snapshot, unless the snapshot is expired. To clean up all of the temporary files after you delete an NDMP snapshot, you must deactivate and reactivate NDMP for the associated NDMP node group.

The deletion of file set snapshots is an I/O intensive operation and the initiation of many of these concurrently might cause contention for I/O resources, resulting in a slowdown of the entire system. In extreme cases, this problem might look like a hung system and some operations might fail. The exact thresholds for this slowdown depend on a number of variables including write and read ratios, I/O capacity, and other concurrent operations. Due to these performance issues, snapshots are deleted from the storage only once per day. The snapshot deletion time is set to 02:30 AM everyday. You can change this value by modifying the **SNAPHOT_DELETE_TIME** parameter that is located in the `/usr/lpp/mmfs/gui/conf/gpfsgui.properties` file. You need to restart the GUI service for the changes to take effect.

# Chapter 24. Creating and managing file clones

A file clone is a writable snapshot of an individual file. File clones can be used to provision virtual machines by creating a virtual disk for each machine by cloning a common base image. A related usage is to clone the virtual disk image of an individual machine as part of taking a snapshot of the machine state.

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. Multiple clones of the same file can be created with no additional space overhead. You can also create clones of clones.

Management of file clones in a GPFS file system includes:



## Creating file clones

File clones can be created from a regular file or a file in a snapshot using the **mmclone** command.

Creating a file clone from a regular file is a two-step process using the **mmclone** command with the **snap** and **copy** keywords:

1. Issue the **mmclone snap** command to create a read-only snapshot of the file to be cloned. This read-only snapshot becomes known as the clone parent. For example, the following command creates a clone parent called **snap1** from the original file **file1**:

   ```
   mmclone snap file1 snap1
   ```

   Alternately, if only one file is specified with the **mmclone snap** command, it will convert the file to a read-only 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. For example, the following command converts **file1** into a clone parent.

   ```
   mmclone snap file1
   ```

2. Issue the **mmclone copy** command to create a writable clone from a clone parent. For example, the following command creates a writable file clone called **file2** from the clone parent **snap1**:

   ```
   mmclone copy snap1 file2
   ```

Creating a file clone where the source is in a snapshot only requires one step using the **mmclone** command with the **copy** keyword. For example, the following command creates a writable file clone called **file3.clone** from a file called **file3** in a snapshot called **snap2**:

```
mmclone copy /fs1/.snapshots/snap2/file3 file3.clone
```

In this case, **file3** becomes the clone parent.

**Note:** Extended attributes of clone parents are not passed along to file clones.

After a clone has been created, the clone and the file that it was cloned from are interchangeable, which is similar to a regular copy (**cp**) command. The file clone will have a new inode number and attributes that can be modified independently of the original file.

Additional clones can be created from the same clone parent by issuing additional **mmclone copy** commands, for example:

```
mmclone copy snap1 file3
```

File clones of clones can also be created, as shown in the following example:

```
mmclone snap file1 snap1
mmclone copy snap1 file2
echo hello >> file2
mmclone snap file2 snap2
mmclone copy snap2 file3
```

The echo command updates the last block of file clone **file2**. When **file2** is snapped to **snap2**, the **mmclone snap** operation is performed as described previously. When a block in **file3** is read, the clone parent inode is found first. For the case of the last block, with the **hello** text, the disk address will be found in **snap2**. However, for other blocks, the disk address will be found in **snap1**.

For complete usage information, see the topic *mmclone command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Listing file clones

Use the **mmclone** command to display status for specified files.

The **show** keyword of the **mmclone** command provides a report to determine the current status of one or more 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.

Consider the following scenario:

1. The **ls** command is issued to show all **img** files in the current directory:

   ```
   ls -ils *.img
   ```

   The system displays output similar to the following:

   ```
   148485 5752576 -rw-r--r-- 1 root root 21474836480 Jan  9 16:19 test01.img
   ```

2. A file clone is then created with the following commands:

   ```
   mmclone snap test01.img base.img
   mmclone copy base.img test02.img
   ```

3. After the file clone is created, the **mmclone show** command is issued 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
   ```

4. A subsequent **ls** command would display output similar to the following:

   ```
   # ls -ils *.img
   148488 5752576 -rw-r--r-- 3 root root 21474836480 Jan  9 16:25 base.img
   148485       0 -rw-r--r-- 1 root root 21474836480 Jan  9 16:19 test01.img
   148480       0 -rw-r--r-- 1 root root 21474836480 Jan  9 16:25 test02.img
   ```

For complete usage information, see the topic *mmclone command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## Deleting file clones

There is no explicit GPFS command available for deleting file clones. File clones can be deleted using a regular delete (**rm**) command. Clone parent files cannot be deleted until all file clone copies of the parent have been deleted and all open file handles to them have been closed.

**Note:** There is a brief period of time, immediately following the deletion of the file clone copies, when deletion of the parent can fail because the clone copy deletions are still running in the background.

## Splitting file clones from clone parents

Use the **mmclone** command to split a file clone from a clone parent.

File clones can be split from their clone parents in one of two ways:

- Using the **mmclone redirect** command to split the file clone from the immediate clone parent only. The clone child remains a file clone, but the clone parent can be deleted.
- Using the **mmclone split** command to split the file clone from all clone parents. This converts the former clone child to a regular file. The clone parent does not change.

For complete usage information, see the topic *mmclone command* in the *IBM Spectrum Scale: Command and Programming Reference*.

## File clones and disk space management

File clones have several considerations that are related to disk space management.

- Replication and storage pools

  Each file clone has its own inode, so attributes and permissions for each clone can be set independently. For example, timestamps (atime, mtime, and ctime) are maintained separately for each clone. Clone parent attributes must be changed separately. If different clones have different values for replication or storage pool, it is not possible for every one of the clones to have all data blocks readable through that clone to be replicated and placed consistent with its replication and pool settings. Thus, changes to replication and storage pool only apply to blocks added to the clone and leave the clone parent unchanged.
- Clone ownership, block counts, and quotas

  Creating a clone parent requires read access to the file being cloned. The person creating the clone parent does not have to be the owner of the original file, but will become the owner of the new clone parent. The block count and disk quota of the original file will be transferred to the new clone parent inode and then set to zero in the original file. Any blocks allocated by copy-on-write of a clone file will be added to the block count in the clone inode, with quota charged against the owner of the file. If even a single byte of data in a clone child is changed, the entire block will be copied from the parent.
- Clones and DMAPI

  Clone parent files and clone copy files will be preserved across migrations and recalls to and from tape.

## File clones and snapshots

When a snapshot is created and a file clone is subsequently updated, the previous state of the file clone will be saved in the snapshot.

When reading a file clone in the snapshot, the system will distinguish between the states of the clone:

- The data block has not been modified since the snapshot was taken, so look for the data in the same file in the next most recent snapshot or active file system.
- The file clone was updated, which indicates that the corresponding data block should be found in the clone parent. The system will look for the data in the clone parent within the same snapshot.

When a snapshot has file clones, those file clones should be deleted or split from their clone parents prior to deleting the snapshot. See “Deleting file clones” on page 357 and “Splitting file clones from clone parents” on page 357 for more information. A policy file can be created to help determine if a snapshot has file clones. See “File clones and policy files” for more information.

## File clones and policy files

Policy files can be created to examine clone attributes.

The following clone attributes can be examined in a policy file:

- The depth of the clone tree.
- If file is an immutable clone parent.
- The fileset ID of the clone parent.
- The inode number of the clone parent for the file.
- If the clone parent is in a snapshot.
- The snapshot ID of the clone parent.

See “File attributes in SQL expressions” on page 308 for more information about the clone attributes available for policy files.

The following example shows a policy file that can be created for displaying clone attributes for all files:

```
RULE EXTERNAL LIST 'x' EXEC ''
RULE 'nonClone' LIST 'x' SHOW('nonclone') WHERE Clone_Parent_Inode IS NULL
RULE 'normalClone' LIST 'x' SHOW(
  'inum ' || varchar(Clone_Parent_Inode) ||
  ' par ' || varchar(Clone_Is_Parent) ||
  ' psn ' || varchar(Clone_Parent_Is_Snap) ||
  ' dep ' || varchar(Clone_Depth))
  WHERE Clone_Parent_Inode IS NOT NULL AND Clone_Parent_Is_Snap == 0
RULE 'snapClone' LIST 'x' SHOW(
  'inum ' || varchar(Clone_Parent_Inode) ||
  ' par ' || varchar(Clone_Is_Parent) ||
  ' psn ' || varchar(Clone_Parent_Is_Snap) ||
  ' dep ' || varchar(Clone_Depth) ||
  ' Fid ' || varchar(Clone_Parent_Fileset_Id) ||
  ' snap ' || varchar(Clone_Parent_Snap_Id))
  WHERE Clone_Parent_Inode IS NOT NULL AND Clone_Parent_Is_Snap != 0
```

If this policy file was called **pol.file**, the following command would display the clone attributes:

```
mmapplypolicy fs0 -P pol.file -I defer -f pol -L 0
```

# Chapter 25. Scale Out Backup and Restore (SOBAR)

Scale Out Backup and Restore (SOBAR) is a specialized mechanism for data protection against disaster only for GPFS file systems that are managed by IBM Spectrum Protect Hierarchical Storage Management (HSM).

**Note:** This feature is available with IBM Spectrum Scale Standard Edition or higher.

To protect a file system against disaster the following steps must be taken to ensure all data is safely stored in a second location:

1. Record the file system configuration with the **mmbackupconfig** command.
2. Ensure all file data is *pre-migrated* (see "Pre-migrating files with external storage pools" on page 334 for more information).
3. Perform a metadata image backup with the **mmimgbackup** command.

The **mmbackupconfig** command must be run prior to running the **mmimgbackup** command. No changes to file system configuration, filesets, quotas, or other settings should be done between running the **mmbackupconfig** command and the **mmimgbackup** command. To recover from a disaster, the **mmrestoreconfig** command must be run prior to running the **mmimgrestore** command. After restoring the image data and adjusting quota settings, the file system can be mounted read-write, and the HSM system re-enabled to permit file data recall. Users may be permitted to access the file system, and/or the system administrator can manually recall file data with the IBM Spectrum Protect HSM command **dsmrecall**.

These commands cannot be run from a Windows node.

## Backup procedure with SOBAR

This section provides a detailed example of the backup procedure used with SOBAR.

Throughout these procedures, the sample file system used is called **smallfs**. Where appropriate, replace this value with your file system name.

1. Backup the cluster configuration information.

   The cluster configuration must be backed up by the administrator. The minimum cluster configuration information needed is: IP addresses, node names, roles, quorum and server roles, cluster-wide configuration settings from **mmchconfig**, cluster manager node roles, remote shell configuration, mutual **ssh** and **rsh** authentication setup, and the cluster UID. More complete configuration information can be found in the **mmsdrfs file** and **CCR**.
2. Preserve disk configuration information.

   Disk configuration must also be preserved in order to recover a file system. The basic disk configuration information needed, for a backup intended for disaster recovery, is the number of disk volumes that were previously available and the sizes of those volumes. In order to recover from a complete file system loss, at least as much disk space as was previously available will be needed for restoration. It is only feasible to restore the image of a file system onto replacement disks if the disk volumes available are of similar enough sizes to the originals that all data can be restored to the new disks. At a minimum, the following disk configuration information is needed:
   - Disk device names
   - Disk device sizes
   - The number of disk volumes
   - NSD server configuration

- Disk RAID configurations
- Failure group designations
- The **mmsdrfs** file contents

3. Backup the GPFS file system configuration information.

   In addition to the disks, the file system built on those volumes has configuration information that can be captured using the **mmbackupconfig** command. This information 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. The file system configuration information can be backed up into a single file using a command similar to the following:

   ```
   mmbackupconfig smallfs -o /tmp/smallfs.bkpcfg.out925
   ```

4. Pre-migrate all newer file data into secondary storage.

   File contents in a space-managed GPFS will reside in secondary storage managed by the HSM. In the case of IBM Spectrum Protect HSM, disk and tape pools will typically hold the offline images of migrated files. HSM can also be used to pre-migrate all newer file data into secondary storage, so that all files will have either a migrated or pre-migrated status (**XATTR**) recorded, and their current contents are copied or updated into the secondary storage. The IBM Spectrum Protect command **dsmmigrate** can be used as follows:

   ```
   dsmmigrate -Premigrate -Recursive /smallfs
   ```

   To optionally check the status of the files that were pre-migrated with the previous command, use the following command:

   ```
   dsmls /smallfs/*
   ```

5. Create a global snapshot of the live file system, to provide a quiescent image for image backup, using a command similar to the following:

   ```
   mmcrsnapshot smallfs smallfssnap
   ```

6. Choose a staging area in which to save the GPFS metadata image files.

   The image backup process stores each piece of the partial file system image backup in its own file in the shared work directory typically used by policy runs. These files can become quite large depending on the number of files in the file system. Also, because the file system holding this shared directory must be accessible to every node participating in the parallel backup task, it might also be a GPFS file system. It is imperative that the staging directory chosen be accessible to both the **tsapolicy** archiver process and the IBM Spectrum Protect Backup-Archive client. This staging directory is specified with the **-g** option of the **mmimgbackup** command.

7. Backup the file system image.

   The following command will back up an image of the GPFS metadata from the file system using a parallel policy run with the default IBM Spectrum Protect backup client to backup the file system metadata image:

   ```
   mmimgbackup smallfs -S smallfssnap -g /u/user/backup -N aixnodes
   ```

   The metadata of the file system, the directories, inodes, attributes, symlinks, and so on are all captured in parallel by using the archive module extension feature of the **mmapplypolicy** command. After completing the parallel execution of the policy-driven archiving process, a collection of image files in this format will remain. These image files are gathered by the **mmimgbackup** command and archived to IBM Spectrum Protect automatically.

   If using the **-N** *nodes* option, it is recommended that the same operating system be used when running **mmimgbackup**. Note the directory created with **-g** *GlobalWorkDirectory* to store the image files.

8. After the image backup is complete, delete the snapshot used for backup with the following command:

   ```
   mmdelsnapshot smallfs smallfssnap
   ```

## Restore procedure with SOBAR

This section provides a detailed example of the restore procedure used with SOBAR.

In order to restore a file system, the configuration data stored from a previous run of **mmbackupconfig** and the image files produced from **mmimgbackup** must be accessible.

Throughout these procedures, the sample file system used is called **smallfs**. Where appropriate, replace this value with your file system name.

1. Restore the metadata image files from **mmimgbackup** and the backup configuration data from **mmbackupconfig** with a **dsmc** command similar to the following:

   ```
   dsmc restore -subdir=yes /u/user/backup/8516/
   ```

2. Retrieve the base file system configuration information.

   Use the **mmrestoreconfig** command to generate a configuration file, which contains the details of the former file system:

   ```
   mmrestoreconfig Device -i InputFile -F QueryResultFile
   ```

3. Recreate NSDs if they are missing.

   Using the output file generated in the previous step as a guide, the administrator might need to recreate NSD devices for use with the restored file system. In the output file, the **NSD configuration** section contains the NSD information; for example:

   ```
   ######## NSD configuration ##################
   ## Disk descriptor format for the mmcrnsd command.
   ## Please edit the disk and desired name fields to match
   ## your current hardware settings.
   ##
   ## The user then can uncomment the descriptor lines and
   ## use this file as input to the -F option.
   #
   # %nsd:
   #   device=DiskName
   #   nsd=nsd8
   #   usage=dataAndMetadata
   #   failureGroup=-1
   #   pool=system
   #
   ```

   If changes are needed, edit the file in a text editor and follow the included instructions to use it as input to the **mmcrnsd** command, then issue the following command:

   ```
   mmcrnsd -F StanzaFile
   ```

4. Recreate the base file system.

   The administrator must recreate the initial file system. The output query file specified in the previous commands can be used as a guide. The following example shows the section of this file that is needed when recreating the file system:

   ```
   ######### File system configuration #############
   ## The user can use the predefined options/option values
   ## when recreating the filesystem.  The option values
   ## represent values from the backed up filesystem.
   #
   # mmcrfs FS_NAME NSD_DISKS -j cluster -k posix -Q yes -L 4194304 --disable-fastea
     -T /fs2 -A no --inode-limit 278016#
   #
   # When preparing the file system for image restore, quota
   # enforcement must be disabled at file system creation time.
   # If this is not done, the image restore process will fail.
   ```

   Do one of the following to recreate the file system:

- Edit the output file. Uncomment the **mmcrfs** command, and specify the appropriate file system name and NSD disk(s). Remove the **-Q** option to ensure quotas are not enabled. Save the changes and run the file as a shell script:

  ```
  sh OutputFile
  ```

- From the command line, issue an **mmcrfs** command similar to the one in the output file, but specify the appropriate file system name and NSD disk(s). Do not specify the **-Q** option to ensure quotas are not enabled.

5. Restore essential file system configuration.

   Using the **mmrestoreconfig** command, the essential file system configuration can be restored to the file system that was just created in the previous step. Quota is disabled in this step because the quota system must remain inactive until after the file system image has been restored. Filesets will also be restored and linked, if necessary, using a method specific for image restore. The **--image-restore** option should be used to restore the configuration data in the proper format for SOBAR; for example:

   ```
   mmrestoreconfig smallfs -i /tmp/smallfs.bkpcfg.out925 --image-restore
   ```

6. Mount the file system in read-only mode for image restore with the following command:

   ```
   mmmount smallfs -o ro
   ```

7. Perform the image restore; for example:

   ```
   mmimgrestore smallfs /u/user/backup/8516/mmPolicy.8551.D4D85229
   ```

8. To optionally display the restored file system structure, use the following command:

   ```
   ls -l /smallfs/*
   ```

   The system displays information similar to the following:

   ```
   -rw-r--r-- 1 root root 1024 Sep 25 11:34 /smallfs/1Kfile.1
   -rw-r--r-- 1 root root 1024 Sep 25 11:34 /smallfs/1Kfile.2
   -rwxr--r-- 1 root root  238 Sep 25 11:34 /smallfs/generateChksums*
   ```

9. Unmount the file system with the following command:

   ```
   mmumount smallfs
   ```

10. Restore quota configuration.

    If any quota enforcement was used in the prior file system, it can be restored now using the **mmrestoreconfig** command. This step will not enable quotas if they were not in use at the time of the configuration backup. To restore the quota configuration, issue a command similar to the following:

    ```
    mmrestoreconfig smallfs -i /tmp/smallfs.bkpcfg.out925 -Q only
    ```

11. Mount the file system in read-write mode with the following command:

    ```
    mmmount smallfs
    ```

12. Delete the unusable HSM directory.

    The **.SpaceMan** directory contains file stubs from the former space management control information. This directory must be deleted prior to restarting HSM management. Use the following command:

    ```
    rm -rf /smallfs/.SpaceMan
    ```

13. To optionally restart HSM, use the following command:

    ```
    dsmmigfs restart
    ```

14. Resume HSM management on the newly reconstructed file system, to resume managing disk space and to permit recall of files, with the following command:

    ```
    dsmmigfs add /smallfs
    ```

15. To optionally display the managed file system from HSM, use the following command:

    ```
    dsmls /smallfs/*
    ```

    All files are currently in the migrate state.

16. To optionally begin recalling files by forcing a specific recall, use the following command:

```
dsmrecall -Recursive /smallfs/*
```

# Chapter 26. Data Mirroring and Replication

The ability to detect and quickly recover from a massive hardware failure is of paramount importance to businesses that make use of real-time data processing systems.

GPFS provides a number of features that facilitate the implementation of highly-available GPFS environments capable of withstanding catastrophic hardware failures. By maintaining a replica of the file system's data at a geographically-separate location, the system sustains its processing using the secondary replica of the file system in the event of a total failure in the primary environment.

On a high level, a disaster-resilient GPFS cluster is made up of two or three, distinct, geographically-separate hardware sites operating in a coordinated fashion. Two of the sites consist of GPFS nodes and storage resources holding a complete replica of the file system. If a third site is active, it consists of a single node and a single disk used as a tiebreaker for GPFS quorum. In the event of a catastrophic hardware failure that disables the operation of an entire site, and assuming the tiebreaker site remains operational, file system services failover to the remaining subset of the cluster and continue serving the data using the replica of the file system that survived the disaster. However, if the tiebreaker fails during the disaster, the remaining number of nodes and disks is insufficient to satisfy the quorum rules and the surviving site loses access to the GPFS file system. A manual procedure is needed to instruct GPFS to disregard the existing quorum assignments and continue operating with whatever resources are available.

The secondary replica is maintained by one of several methods:

- Synchronous mirroring utilizing GPFS replication.

  The data and metadata replication features of GPFS are used to implement synchronous mirroring between a pair of geographically-separate sites. The use of logical replication-based mirroring offers a generic solution that relies on no specific support from the disk subsystem beyond the basic ability to read and write data blocks..

- Synchronous mirroring utilizing storage-based replication.

  Hardware replication creates persistent mirroring relationship between pairs of Logical Units (LUNs) on two subsystems connected over SAN or LAN links. All updates performed on the set of primary, source, or LUNs appear in the same order on the secondary, target, or disks in the target subsystem. Hardware replication provides for an exact bitwise replica of the content of the source as seen at the time of the failure on the target if the source volume fails. A range of technologies can be used to provide synchronous replication such as Metro Mirror on DS8000® or Storwize® or Synchronous Remote Mirroring on XIV®.

- Asynchronous mirroring utilizing GPFS-based replication.

  Asynchronous replication functionality provides a similar crash consistent copy of data as synchronous replication but in normal operation the secondary copy of data will lag behind the primary by some period of time. For more information, see the topic *Introduction to AFM-based Asynchronous Disaster Recovery (AFM DR)* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

- Asynchronous mirroring utilizing storage-based replication.

  Asynchronous replication functionality provides a similar crash consistent copy of data as synchronous replication but in normal operation the secondary copy of data will lag behind the primary by some time. A range of technologies can be used to provide asynchronous replication such as Global Mirror on DS8000 or Storwize or Asynchronous Remote Mirroring on XIV.

- Point in time copy using storage-based functionality.

  Periodic point-in-time copies of the file system are taken using the functionality such as FlashCopy® on the DS8000 or Storwize or Snapshot on XIV. This copy could be used as a source of a complete file

system consistent backup to be taken to a remote site or could be used in conjunction with other replication capabilities to use for isolated testing of disaster recovery procedures.

The primary advantage of both synchronous mirroring methods is the minimization of the risk of permanent data loss. Both methods provide two consistent, up-to-date replicas of the file system, each available for recovery if the other one fails. However, inherent to all solutions that synchronously mirror data over a wide area network link is the latency penalty that is induced by the replicated write I/Os. This makes both synchronous mirroring methods prohibitively inefficient for certain types of performance-oriented applications of where there is a longer distance between sites. The asynchronous method effectively eliminates this penalty but in a situation where the primary site is lost, there might be updates that have not yet been transferred to the secondary site. Asynchronous replication will still provide a crash consistent and restartable copy of the primary data.

## General considerations for using storage replication with GPFS

This topic describes the general considerations that need to be followed for using storage replication with IBM Spectrum Scale.

Different storage-level replication capabilities are available on both IBM and non-IBM storage systems. IBM provides storage-level replication functionality on the following platforms:

- The DS8000 provides synchronous replication with Metro Mirror and asynchronous replication with Global Mirror. Three and four site replication topologies are also possible by combining these functions. For more information, see **IBM DS8000 series V7.2 documentation** at http://www-01.ibm.com/support/knowledgecenter/HW213_7.2.0/com.ibm.storage.ssic.help.doc/f2c_ichomepage.htm.
- The Storwize family of storage systems also provides a synchronous replication capability with Metro Mirror and has two versions of asynchronous replication called Global Mirror and Global Mirror with Change Volumes. Point in Time copy functionality is provided by FlashCopy. For more information, see **IBM Storwize V7000** at http://www-01.ibm.com/support/knowledgecenter/ST3FR7/welcome.
- The XIV provides both synchronous and asynchronous Remote Replication and also provides point in time copy functionality referred to as Snapshot. For more information, see **IBM XIV Storage System documentation** at http://www-01.ibm.com/support/knowledgecenter/STJTAG/com.ibm.help.xivgen3.doc/xiv_kcwelcomepage.html.

**Note:** In this document, synchronous replication is referred to as Metro Mirror, asynchronous replication is referred to as Global Mirror, and point in time copy functionality is referred to as FlashCopy.

## Data integrity and the use of consistency groups

This topic provides description about the data integrity issues and the use of consistency groups to recover data.

The integrity of the post-disaster replica of the file system that is contained on the replication target volumes depends on the assumption that the order of dependent write operations is preserved by the mirroring mechanism. Replication is often referred to as providing a crash consistent copy of the replicated data.

When a synchronous replication environment is running a secondary copy that is identical to the primary, if a situation such as a sudden loss of the primary servers happens, then the secondary servers would remain consistent. However, for an environment with multiple replicated LUNs or multiple storage systems, ensure that a rolling disaster does not compromise the consistency of the replicated data.

Figure 10 on page 367 illustrates this problem. Consider a setup with two primary and two secondary subsystems, each providing access to two LUNs. The four primary LUNs make up the primary replica of the file system. At some point, GPFS attempts to issue a sequence of four write requests, one to each primary disk, with the expectation that the updates appear in the exact order they were issued. If PPRC

path 1 breaks before the start of the first write, the recovery site receives updates 3 and 4, but not necessarily 1 and 2. This is a result that violates the write dependency rule and renders the target replica of the file system unusable.

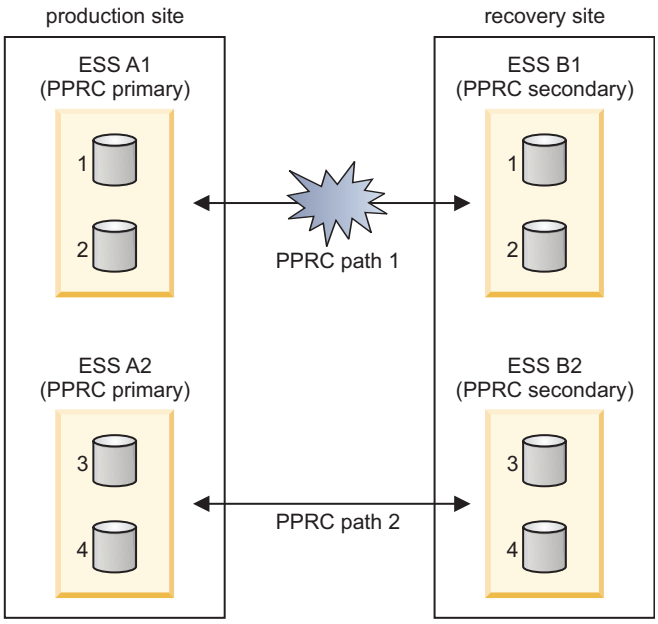



*Figure 10. Violation of write ordering without the use of consistency group functionality*

The solution to this issue is the functionality that ensures that if one of the replicated devices suspends, then all other devices suspend in a consistent manner, preserving the write ordering on the secondary devices. This is often referred to as consistency group functionality or the definition of a consistency group for a set of volumes. For asynchronous replication, the issue is more fundamental in that the secondary copy of data lags behind the primary and so it is required that the replication mechanism enables a recovery to the same point in time for all volumes involved in the replication.

GPFS™ relies on log recovery techniques to provide for the atomicity of updates to metadata of the file system, which is why such behavior would expose GPFS to a serious data integrity risk. Therefore, to provide for the proper ordering of updates to the recovery copy of the file system, it is required that all devices that are related to a GPFS cluster are included in a single consistency group or replication session.

## Handling multiple versions of IBM Spectrum Scale data

This topic provides description on handling multiple versions of data in IBM Spectrum Scale.

The primary copy of a GPFS file system and a hardware replicated copy cannot coexist in the same GPFS cluster. A node can mount either the original copy of the file system or one of its replicas, but not both. This restriction has to do with the current implementation of the NSD-to-LUN mapping mechanism, which scans all locally attached disks, searching for a specific value (the NSD ID) at a particular location on disk. If both the original volume and a hardware replica are visible to a particular node, these disks would appear to GPFS as distinct devices with identical NSD IDs.

For this reason, users are asked to zone their SAN configurations such that at most one replica of any given GPFS disk is visible from any node. That is, the nodes in your production cluster should have access to the disks that make up the actual file system but should not see the disks that hold the replicated copies, whereas the backup server should see the replication targets but not the originals.

Alternatively, you can use the **nsddevices** user exit located in /var/mmfs/etc/ to explicitly define the subset of the locally visible disks to be accessed during the NSD device scan on the local node.

The following procedure is used to define an **nsddevices** user exit file to instruct GPFS to use a specific disk **diskA1** rather than other copies of this device, which might also be available:

```
echo "echo diskA1 hdisk" > /var/mmfs/etc/nsddevices chmod 744 /var/mmfs/etc/nsddevices
```

Refer to the prolog of /usr/lpp/mmfs/samples/nsddevices.samples for detailed instructions on the usage of **nsddevices**.

# Continuous Replication of IBM Spectrum Scale data

This topic provides a brief description on replication of IBM Spectrum Scale data.

## Synchronous mirroring with GPFS replication

In a configuration utilizing GPFS replication, a single GPFS cluster is defined over three geographically-separate sites consisting of two production sites and a third tiebreaker site. One or more file systems are created, mounted, and accessed concurrently from the two active production sites.

The data and metadata replication features of GPFS are used to maintain a secondary copy of each file system block, relying on the concept of disk failure groups to control the physical placement of the individual copies:

1. Separate the set of available disk volumes into two failure groups. Define one failure group at each of the active production sites.
2. Create a replicated file system. Specify a replication factor of 2 for both data and metadata.

When allocating new file system blocks, GPFS always assigns replicas of the same block to distinct failure groups. This provides a sufficient level of redundancy allowing each site to continue operating independently should the other site fail.

GPFS enforces a node quorum rule to prevent multiple nodes from assuming the role of the file system manager in the event of a network partition. Thus, a majority of quorum nodes must remain active in order for the cluster to sustain normal file system usage. Furthermore, GPFS uses a quorum replication algorithm to maintain the content of the file system descriptor (one of the central elements of the GPFS metadata). When formatting the file system, GPFS assigns some number of disks (usually three) as the descriptor replica holders that are responsible for maintaining an up-to-date copy of the descriptor. Similar to the node quorum requirement, a majority of the replica holder disks must remain available at all times to sustain normal file system operations. This file system descriptor quorum is internally controlled by the GPFS daemon. However, when a disk has failed due to a disaster you must manually inform GPFS that the disk is no longer available and it should be excluded from use.

Considering these quorum constraints, it is suggested that a third site in the configuration fulfill the role of a tiebreaker for the node and the file system descriptor quorum decisions. The tiebreaker site consists of:

1. A single quorum node

   As the function of this node is to serve as a tiebreaker in GPFS quorum decisions, it does not require normal file system access and SAN connectivity. To ignore disk access errors on the tiebreaker node, enable the **unmountOnDiskFail** configuration parameter through the **mmchconfig** command. When enabled, this parameter forces the tiebreaker node to treat the lack of disk connectivity as a local error, resulting in a failure to mount the file system, rather that reporting this condition to the file system manager as a disk failure.
2. A single network shared disk

The function of this disk is to provide an additional replica of the file system descriptor file needed to sustain quorum should a disaster cripple one of the other descriptor replica disks. Create a network shared disk over the tiebreaker node's internal disk defining:

- the local node as an NSD server
- the disk usage as **descOnly**

  The **descOnly** option instructs GPFS to only store file system descriptor information on the disk.

This three-site configuration is resilient to a complete failure of any single hardware site. Should all disk volumes in one of the failure groups become unavailable, GPFS performs a transparent failover to the remaining set of disks and continues serving the data to the surviving subset of nodes with no administrative intervention. While nothing prevents you from placing the tiebreaker resources at one of the active sites, to minimize the risk of double-site failures it is suggested you install the tiebreakers at a third, geographically distinct location.

**Note:** There are no special networking requirements for this configuration. For example:

- You do not need to create different subnets.
- You do not need to have GPFS nodes in the same network across the two production sites.
- The production sites can be on different virtual LANs (VLANs).

The high-level organization of a replicated GPFS cluster for synchronous mirroring where all disks are directly attached to all nodes in the cluster is shown in Figure 11 on page 370. An alternative to this design would be to have the data served through designated NSD servers.

With GPFS release 4.1.0, a new, more fault-tolerant configuration mechanism has been introduced as the successor for the server-based mechanisms. The server-based configuration mechanisms consist of two configuration servers specified as the primary and secondary cluster configuration server. The new configuration mechanism uses all specified quorum nodes in the cluster to hold the GPFS configuration and is called CCR (Clustered Configuration Repository). The CCR is used by default during cluster creation unless the CCR is explicitly disabled. The mmlscluster command reports the configuration mechanism in use in the cluster.

The following sections describe the differences regarding disaster recovery for the two configuration mechanisms.

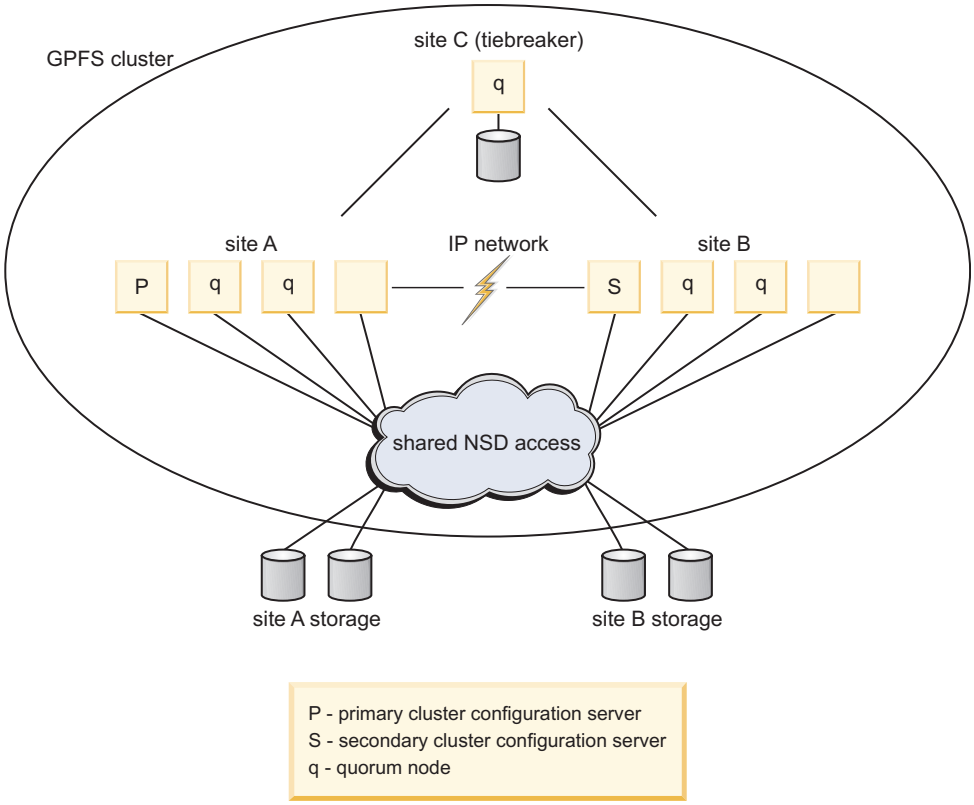



*Figure 11. Synchronous mirroring utilizing GPFS replication*

## Setting up an IBM Spectrum Scale cluster with synchronous mirroring utilizing Spectrum Scale replication

The setting up of a GPFS cluster by utilizing replication can be described through an example.

To establish a disaster-resilient GPFS cluster utilizing replication as shown in Figure 11, consider the configuration:

**Site A** Consisting of:

- Nodes – **nodeA001**, **nodeA002**, **nodeA003**, **nodeA004**
- Disk device names – **diskA1**, **diskA2**

  **diskA1** and **diskA2** are SAN-attached and accessible from all nodes at site **A** and site **B**.

**Site B** Consisting of:

- Nodes – **nodeB001**, **nodeB002**, **nodeB003**, **nodeB004**
- Disks – **diskB1**, **diskB2**

  **diskB1** and **diskB2** are SAN-attached and accessible from all nodes at site **A** and site **B**.

**Site C (tiebreaker)**

Consisting of:

- Node – **nodeC**
- Disk – **diskC**

  **diskC** is an NSD defined over the internal disk of the node **nodeC** and is directly accessible only from site **C**

1. Create a GPFS cluster selecting **nodeA001** at site A as the primary cluster data server node, **nodeB001** at site **B** as the secondary cluster data server nodes, and the nodes in the cluster contained in the file **clusterNodes**. The **clusterNodes** file contains the node descriptors:

   ```
   nodeA001:quorum-manager
   nodeA002:quorum-manager
   nodeA003:quorum-manager
   nodeA004:client
   nodeB001:quorum-manager
   nodeB002:quorum-manager
   nodeB003:quorum-manager
   nodeB004:client
   nodeC:quorum-client
   ```

   Issue this command:

   ```
   mmcrcluster –N clusterNodes
   ```

   This cluster is created using the new Clustered Configuration Repository (CCR) as its configuration mechanisms. All specified quorum nodes contain the GPFS configuration. The cluster can be created using the traditional, server-based configuration mechanisms by issuing the following command instead:

   ```
   mmcrcluster –N clusterNodes --ccr-disable –p nodeA001 –s nodeB001
   ```

2. Prevent false disk errors in the SAN configuration from being reported to the file system manager by enabling the **unmountOnDiskFail** option on the tiebreaker node:

   ```
   mmchconfig unmountOnDiskFail=yes -N nodeC
   ```

3. Define the set of network shared disks for the cluster where disks at sites **A** and **B** are assigned to failure groups 1 and 2, respectively. The tiebreaker disk is assigned to failure group 3. The disk descriptors contained in the file **clusterDisks** are:

   ```
   %nsd: device=/dev/diskA1
       servers=nodeA002,nodeA003
       usage=dataAndMetadata
       failureGroup=1


   %nsd: device=/dev/diskA2
       servers=nodeA003,nodeA002
       usage=dataAndMetadata
       failureGroup=1


   %nsd: device=/dev/diskB1
       servers=nodeB002,nodeB003
       usage=dataAndMetadata
       failureGroup=2


   %nsd: device=/dev/diskB2
       servers=nodeB003,nodeB002
       usage=dataAndMetadata
       failureGroup=2


   %nsd: device=/dev/diskC1
       servers=nodeC
       usage=descOnly
       failureGroup=3
   ```

   Issue this command:

   ```
   mmcrnsd –F clusterDisks
   ```

4. Issue the **mmlsnsd** command to verify that the network shared disks have been created:

   ```
   mmlsnsd -m
   ```

Output is similar to this:

```
Disk name    NSD volume ID      Device         Node name         Remarks
---------------------------------------------------------------------------------
 gpfs1nsd     0972445B416BE502   /dev/diskA1    nodeA002          server node
 gpfs1nsd     0972445B416BE502   /dev/diskA1    nodeA003          server node
 gpfs2nsd     0972445B416BE509   /dev/diskA2    nodeA002          server node
 gpfs2nsd     0972445B416BE509   /dev/diskA2    nodeA003          server node
 gpfs3nsd     0972445F416BE4F8   /dev/diskB1    nodeB002          server node
 gpfs3nsd     0972445F416BE4F8   /dev/diskB1    nodeB003          server node
 gpfs4nsd     0972445F416BE4FE   /dev/diskB2    nodeB002          server node
 gpfs4nsd     0972445F416BE4FE   /dev/diskB2    nodeB003          server node
 gpfs5nsd     0972445D416BE504   /dev/diskC1    nodeC             server node
```

5. Start the GPFS daemon on all nodes:

```
mmstartup -a
```

6. Create a replicated file system **fs0**:

```
mmcrfs /gpfs/fs0 fs0 –F clusterDisks –m 2 –M 2 –r 2 –R 2
```

7. Mount **fs0** on all nodes at sites A and B.

## Steps to take after a disaster when using Spectrum Scale replication

Utilizing GPFS replication allows for *failover* to the surviving site without disruption of service as long as both the remaining site and the tiebreaker site remain functional. It remains in this state until a decision is made to restore the operation of the affected site by executing the *failback* procedure. If the tiebreaker site is also affected by the disaster and is no longer operational, GPFS quorum is broken and manual intervention is required to resume file system access.

Existing quorum designations must be relaxed in order to allow the surviving site to fulfill quorum requirements:

1. To relax node quorum, temporarily change the designation of each of the failed quorum nodes to non-quorum nodes. Issue the **mmchnode --nonquorum** command.
2. To relax file system descriptor quorum, temporarily eliminate the failed disks from the group of disks from which the GPFS daemon uses to write the file system descriptor file to. Issue the **mmfsctl exclude** command for each of the failed disks.

While the GPFS cluster is in a failover state, it is suggested that no changes to the GPFS configuration be made. If the server-based configuration mechanism is in use, changes to your GPFS configuration require both cluster configuration servers to be operational. If both servers are not operational, the sites would have distinct, and possibly inconsistent, copies of the GPFS **mmsdrfs** configuration data file. While the servers can be migrated to the surviving site, it is best to avoid this step if the disaster does not leave the affected site permanently disabled.

If it becomes absolutely necessary to modify the GPFS configuration while in failover mode, for example to relax quorum, you must ensure that all nodes at the affected site are powered down and left in a stable inactive state. They must remain in such state until the decision is made to execute the failback procedure. As a means of precaution, we suggest disabling the GPFS autoload option on all nodes to prevent GPFS from bringing itself up automatically on the affected nodes should they come up spontaneously at some point after a disaster.

**Failover to the surviving site:**

Following a disaster, which failover process is implemented depends upon whether or not the tiebreaker site is affected.

**Failover without the loss of tiebreaker site C**

The proposed three-site configuration is resilient to a complete failure of any single hardware site. Should all disk volumes in one of the failure groups become unavailable, GPFS performs a transparent failover to

| the remaining set of disks and continues serving the data to the surviving subset of nodes with no
| administrative intervention.

| **Failover with the loss of tiebreaker site C with server-based configuration in use**

| If both site **A** and site **C** fail:

| 1. Shut the GPFS daemon down on the surviving nodes at site **B**, where the file **gpfs.siteB** lists all of the
| nodes at site **B**:
| `mmshutdown -N gpfs.siteB`
| 2. If it is necessary to make changes to the configuration, migrate the primary cluster configuration
| server to a node at site **B**:
| `mmchcluster -p nodeB002`
| 3. Relax node quorum by temporarily changing the designation of each of the failed quorum nodes to
| non-quorum nodes:
| `mmchnode --nonquorum -N nodeA001,nodeA002,nodeA003,nodeC`
| 4. Relax file system descriptor quorum by informing the GPFS daemon to migrate the file system
| descriptor off of the failed disks:
| `mmfsctl fs0 exclude -d "gpfs1nsd;gpfs2nsd;gpfs5nsd"`
| 5. Restart the GPFS daemon on the surviving nodes:
| `mmstartup -N gpfs.siteB`
| 6. Mount the file system on the surviving nodes at site B.

| **Failover with the loss of tiebreaker site C with Clustered Configuration Repository (CCR) in use**

| If both site A and site C fail:

| 1. Shut the GPFS daemon down on the surviving nodes at site B , where the file gpfs.siteB lists all of the
| nodes at site B :
| `mmdsh -N gpfs.siteB /usr/lpp/mmfs/bin/mmshutdown`
| 2. Changing (downgrading) the quorum assignments when half or more of the quorum nodes are no
| longer available at site B using the `–- force` option :
| `mmchnode –-nonquorum -N nodeA001,nodeA002,nodeA003,nodeC --force`
| 3. Relax file system descriptor quorum by informing the GPFS daemon to migrate the file system
| descriptor off of the failed disks:
| `mmfsctl fs0 exclude -d "gpfs1nsd;gpfs2nsd;gpfs5nsd"`
| 4. Restart the GPFS daemon on the surviving nodes:
| `mmstartup -N gpfs.siteB`
| 5. Mount the file system on the surviving nodes at site B.

| Make no further changes to the quorum designations at site B until the failed sites are back on line and
| the following failback procedure has been completed.

| Do not shut down the current set of nodes on the surviving site B and restart operations on the failed
| sites A and C. This will result in a non-working cluster.

| **Failback procedures:**

| Which failback procedure you follow depends upon whether the nodes and disks at the affected site have
| been repaired or replaced.

| If the disks have been repaired, you must also consider the state of the data on the failed disks:

| • For nodes and disks that have been repaired and *you are certain* the data on the failed disks has not
| been changed, follow either:

- *failback with temporary loss and no configuration changes*
- *failback with temporary loss and configuration changes*

- If the nodes have been replaced and either the disks have been replaced or repaired, and *you are not certain* the data on the fail disks has not been changed, follow the procedure for *failback with permanent loss*.

**Delayed failures:** In certain failure cases the loss of data may not be immediately apparent. For example, consider this sequence of events:

1. Site **B** loses connectivity with sites **A** and **C**.
2. Site **B** then goes down due to loss of node quorum.
3. Sites **A** and **C** remain operational long enough to modify some of the data on disk but suffer a disastrous failure shortly afterwards.
4. Node and file system descriptor quorums are overridden to enable access at site **B**.

Now the two replicas of the file system are inconsistent and the only way to reconcile these copies during recovery is to:

1. Remove the damaged disks at sites **A** and **C**.
2. Either replace the disk and format a new NSD or simply reformat the existing disk if possible.
3. Add the disk back to the file system, performing a full resynchronization of the file system's data and metadata and restore the replica balance using the **mmrestripefs** command.

*Failback with temporary loss and no configuration changes:*

If the outage was of a temporary nature and your configuration has not been altered, it is a simple process to fail back to the original state.

After all affected nodes and disks have been repaired and *you are certain* the data on the failed disks has not been changed:

1. Start GPFS on the repaired nodes where the file **gpfs.sitesAC** lists all of the nodes at sites **A** and **C**:
   ```
   mmstartup -N gpfs.sitesAC
   ```
2. Restart the affected disks. If more than one disk in the file system is down, they must all be started at the same time:
   ```
   mmchdisk fs0 start -a
   ```

*Failback with temporary loss and configuration changes in the server-based configuration:*

If the outage was of a temporary nature and your configuration has been altered, follow this procedure to fail back to the original state in case primary and secondary configuration servers are in use.

After all affected nodes and disks have been repaired and *you are certain* that the data on the failed disks has not been changed:

1. Ensure that all nodes have the latest copy of the `mmsdrfs` file:
   ```
   mmchcluster -p LATEST
   ```

   For more information about the `mmsdrfs` file, see *Recovery from loss of GPFS cluster configuration data file* in the *IBM Spectrum Scale: Problem Determination Guide*.
2. Migrate the primary cluster configuration server back to site **A**:
   ```
   mmchcluster -p nodeA001
   ```
3. Restore node quorum designations at sites **A** and **C**:
   ```
   mmchnode --quorum -N nodeA001,nodeA002,nodeA003,nodeC
   ```
4. Start GPFS on the repaired nodes where the file **gpfs.sitesAC** lists all of the nodes at sites **A** and **C**:
   ```
   mmstartup -N gpfs.sitesAC
   ```

5. Restore the file system descriptor quorum by informing the GPFS to include the repaired disks:

```
mmfsctl fs0 include -d "gpfs1nsd;gpfs2nsd;gpfs5nsd"
```

6. Bring the disks online and restripe the file system across all disks in the cluster to restore the initial replication properties:

```
mmchdisk fs0 start -a
mmrestripefs fs0 -b
```

The **-r** flag may be used on the **mmrestripefs** command instead.

*Failback with temporary loss using the Clustered Configuration Repository (CCR) configuration mechanism:*

If the outage was of a temporary nature and your configuration has been altered, follow this procedure to failback to the original state, in case the Clustered Configuration Repository (CCR) configuration scheme is in use.

After all affected nodes and disks have been repaired and you are certain the data on the failed disks has not been changed, complete the following steps.

1. Shut down the GPFS daemon on the surviving nodes at site B , and on the former failed and now recovered sites A and C , where the file `gpfs.siteB` lists all of the nodes at site B and the file `gpfs.siteA` lists all of the nodes at site A and the tiebreaker node at site C:

```
mmshutdown -N gpfs.siteB
mmshutdown -N gpfs.siteA
mmshutdown -N nodeC
```

2. Restore original node quorum designation for the tiebreaker site C at site B and start GPFS on site C:

```
mmstartup -N gpfs.siteB
mmchnode --quorum -N nodeC
mmstartup -N nodeC
```

3. Restore original node quorum designation for site A at site B and start GPFS on site A:

```
mmchnode --quorum -N nodeA001,nodeA002,nodeA003
mmstartup -N gpfs.siteA
```

4. Restore the file system descriptor quorum by informing the GPFS to include the repaired disks:

```
mmumount fs0 -a;mmfsctl fs0 include -d "gpfs1nsd;gpfs2nsd;gpfs5nsd"
```

5. Mount the file system on all nodes at sites A and B.

**Note:** Do not allow the failed sites A and C to come online at the same time or when site B is unavailable or not functional.

6. Bring the disks online and restripe the file system across all disks in the cluster to restore the initial replication properties:

```
mmchdisk fs0 start -a
mmrestripefs fs0 -b
```

The `-r` flag can be used on the `mmrestripefs` command instead.

*Failback with permanent loss:*

If an outage is of a permanent nature, follow steps to remove and replace the failed resources, and then resume the operation of GPFS across the cluster.

1. Remove the failed resources from the GPFS configuration
2. Replace the failed resources, then add the new resources into the configuration
3. Resume the operation of GPFS across the entire cluster

Assume that sites **A** and **C** have had permanent losses. To remove all references of the failed nodes and disks from the GPFS configuration and replace them:

**Procedure when the server-based configuration scheme is in use**

1. To remove the failed resources from the GPFS configuration:
   a. If as part of the failover process, *you did not* migrate the primary cluster configuration server, migrate the server to node **nodeB002** at site B:
      ```
      mmchcluster –p nodeB002
      ```
   b. Delete the failed disks from the GPFS configuration:
      ```
      mmdeldisk fs0 "gpfs1nsd;gpfs2nsd;gpfs5nsd"
      mmdelnsd "gpfs1nsd;gpfs2nsd;gpfs5nsd"
      ```
   c. Delete the failed nodes from the GPFS configuration:
      ```
      mmdelnode -N nodeA001,nodeA002,nodeA003,nodeA004,nodeC
      ```
2. If there are new resources to add to the configuration:
   a. Add the new nodes at sites **A** and **C** to the cluster where the file **gpfs.sitesAC** lists of the new nodes:
      ```
      mmaddnode -N gpfs.sitesAC
      ```
   b. Ensure that all nodes have the latest copy of the **mmsdrfs** file:
      ```
      mmchcluster -p LATEST
      ```
   c. Migrate the primary cluster configuration server back to site **A**:
      ```
      mmchcluster -p nodeA001
      ```
   d. Start GPFS on the new nodes
      ```
      mmstartup -N gpfs.sitesAC
      ```
   e. Prepare the new disks for use in the cluster, create the NSDs using the original disk descriptors for site **A** contained in the file **clusterDisksAC**:
      ```
      %nsd: device=/dev/diskA1
          servers=nodeA002,nodeA003
          usage=dataAndMetadata
          failureGroup=1

      %nsd: device=/dev/diskA2
          servers=nodeA003,nodeA002
          usage=dataAndMetadata
          failureGroup=1

      %nsd: device=/dev/diskC1
          servers=nodeC
          usage=descOnly

      failureGroup=3mmcrnsd -F clusterDisksAC
      ```
   f. Add the new NSDs to the file system specifying the **-r** option to rebalance the data on all disks:
      ```
      mmadddisk fs0 -F clusterDisksAC -r
      ```

**Procedure when Clustered Configuration Repository (CCR) is in use**

1. To remove the failed resources from the GPFS configuration:
   a. Delete the failed disks from the GPFS configuration:
      ```
      mmdeldisk fs0 "gpfs1nsd;gpfs2nsd;gpfs5nsd"
      mmdelnsd "gpfs1nsd;gpfs2nsd;gpfs5nsd"
      ```
   b. Delete the failed nodes from the GPFS configuration:
      ```
      mmdelnode -N nodeA001,nodeA002,nodeA003,nodeA004,nodeC
      ```
2. If there are new resources to add to the configuration:
   a. Add the new nodes at sites **A** and **C** to the cluster where the file **gpfs.sitesAC** lists of the new nodes:
      ```
      mmaddnode -N gpfs.sitesAC
      ```
   b. Restore original quorum node assignments at site B:

```
mmchnode --quorum -N nodeA001,nodeA002,nodeA003,nodeC
```

c. Start GPFS on the new nodes

```
mmstartup -N gpfs.sitesAC
```

d. Prepare the new disks for use in the cluster, create the NSDs using the original disk descriptors for site **A** contained in the file **clusterDisksAC**:

```
%nsd: device=/dev/diskA1
 servers=nodeA002,nodeA003
 usage=dataAndMetadata
 failureGroup=1

 %nsd: device=/dev/diskA2
 servers=nodeA003,nodeA002
 usage=dataAndMetadata
 failureGroup=1

 %nsd: device=/dev/diskC1
 servers=nodeC
 usage=descOnly

 failureGroup=3mmcrnsd -F clusterDisksAC
```

e. Add the new NSDs to the file system specifying the **-r** option to rebalance the data on all disks:

```
mmadddisk fs0 -F clusterDisksAC -r
```

## Synchronous mirroring utilizing storage based replication

This topic describes synchronous mirroring utilizing storage-based replication.

Synchronous replication in the storage layer continuously updates a secondary (target) copy of a disk volume to match changes made to a primary (source) volume. A pair of volumes are configured in a replication relationship, during which all write operations performed on the source are synchronously mirrored to the target device.

The synchronous replication protocol guarantees that the secondary copy is constantly up-to-date by ensuring that the primary copy is written only if the primary storage subsystem received an acknowledgment that the secondary copy has been written. The paired volumes typically reside on two distinct and geographically separated storage systems communicating over a SAN or LAN link.

Most synchronous replication solutions provide a capability to perform an incremental resynchronization of data when switching between primary and secondary storage systems. After the failure of the primary volume (or the failure of the entire storage subsystem or site), users perform a failover, which suspends the relationship between the given pair of volumes and turns the target volume into a primary. When a volume enters the suspended state, a modification bitmap is established to keep track of the write operations performed on that volume to allow for an efficient resynchronization.

Once the operation of the original primary volume has been restored, a failback is executed to resynchronize the content of the two volumes. The original source volume is switched to the target mode, after which all modified data tracks (those recorded in the modification bitmap) are copied from the original target disk. The volume pair can then be suspended again and a similar process performed to reverse the volumes' roles, thus bringing the pair into its initial state.

A GPFS cluster using hardware-based replication can be established in two manners:

- A single GPFS cluster encompassing two sites and an optional tiebreaker site
- Two distinct GPFS clusters

### An active-active Spectrum Scale cluster

In an active-active cluster, a single GPFS cluster contains two active sites and an optional tiebreaker site.

The high-level organization of an active/active GPFS cluster using hardware replication is illustrated in Figure 12.A single GPFS cluster is created over three sites. The data is mirrored between two active sites with a cluster configuration server residing at each site and a tiebreaker quorum node installed at the third location. The presence of an optional tiebreaker node allows the surviving site to satisfy the node quorum requirement with no additional intervention. Without the tiebreaker, the failover procedure requires an additional administrative command to relax node quorum and allow the remaining site to function independently. Furthermore, the nodes at the recovery site have direct disk paths to the primary site's storage.

The GPFS configuration resides either on the two configuration server (primary and secondary), when the cluster has been created with the Clustered Configuration Repository (CCR) disable option (**mmcrcluster**), or on each quorum node, when the cluster has Clustered Configuration Repository (CCR) enabled, or on the primary/secondary, when the Clustered Configuration Repository (CCR) is disabled.

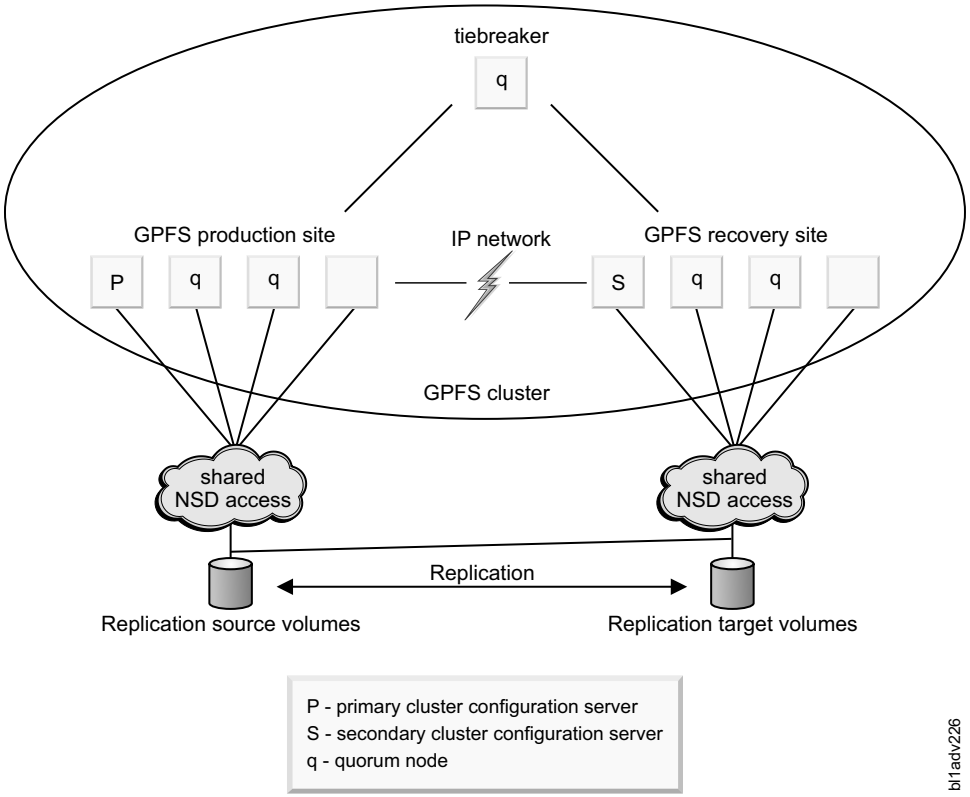



*Figure 12. A synchronous active-active replication-based mirrored GPFS configuration with a tiebreaker site*

**Setting up an active-active GPFS configuration:**

This example demonstrates how to configure an active-active GPFS cluster.

To establish an active-active GPFS cluster using hardware replication with a tiebreaker site as shown in Figure 12, consider the configuration:

**Site A (production site)**
Consists of:

- Nodes – **nodeA001**, **nodeA002**, **nodeA003**, **nodeA004**
- Storage subsystems – **A**
- Disk volumes – **diskA** on storage system **A**
  **diskA** is SAN-attached and accessible from sites **A** and **B**

**Site B (recovery site)**
Consists of:

- Nodes – **nodeB001**, **nodeB002**, **nodeB003**, **nodeB004**
- Storage subsystems – **B**
- Disk volumes – **diskB** on storage system **B**

  **diskB** is SAN-attached and accessible from site **B** only

**Site C (tiebreaker)**
Consists of:

- Nodes – **nodeC**

  **diskC** is an NSD defined over the internal disk of the node **nodeC** and is directly accessible only from site **C**

1. Establish a hardware replication connectivity between the storage systems and then establish the synchronous replication volume pair between the source and target using the copy entire volume option. In this case, it would be **diskA–diskB**.
2. In order to protect the order of dependent writes that span multiple disk volumes, multiple storage systems, or both, the consistency group functionality of the storage system should be used with all GPFS devices in the same consistency group.
3. Create a GPFS cluster defining the primary cluster configuration server as nodes **nodeA001** at site **A**, the secondary cluster configuration server as **nodeB001** at site **B**, an equal number of quorum nodes at each site, including the tiebreaker node at site **C**, **nodeC**. To prevent the tiebreaker node from assuming the role of file system manager, define it as **client**. Define all other quorum nodes as **manager**. List the nodes in the cluster in the file **NodeDescFile**. The **NodeDescFile** file contains the node descriptors:

   ```
   nodeA001:quourum-manager
   nodeA002:quorum-manager
   nodeA003:quorum-manager
   nodeA004:client
   nodeB001:quorum-manager
   nodeB002:quorum-manager
   nodeB003:quorum-manager
   nodeB004:client
   nodeC:quorum-client
   ```

   Issue this command:

   ```
   mmcrcluster –N NodeDescFile –p nodeA001 –s nodeB001
   ```
4. Enable the **unmountOnDiskFail** option on the tiebreaker node preventing false disk errors in the SAN configuration from being reported to the file system manager by issuing the **mmchconfig** command:

   ```
   mmchconfig unmountOnDiskFail=yes -N nodeC
   ```
5. Create an NSD over **diskA**. The disk descriptor contained in the file **DiskDescFile** is:

   ```
   /dev/diskA:nodeA001:nodeA002:dataAndMetadata:1
   ```

   Issue this command:

   ```
   mmcrnsd –F DiskDescFileP
   ```
6. Start the GPFS daemon on all nodes:

   ```
   mmstartup -a
   ```
7. Create a GPFS file system and mount it on all nodes at sites **A** and **B**.

   ```
   mmcrfs /gpfs/fs0 fs0 -F DiskDescFile
   ```

**Failover to the recovery site and subsequent failback for an active/active configuration:**

For an active/active storage replication based cluster, complete these steps to restore access to the file system through site **B** after site **A** has experienced a disastrous failure.

**Procedure when the server-based configuration scheme is in use**

1. Stop the GPFS daemon on the surviving nodes as site **B** where the file **gpfs.siteB** lists all of the nodes at site **B**:

   ```
   mmdsh -N gpfs.siteB /usr/lpp/mmfs/bin/mmshutdown
   ```

2. Perform the appropriate commands to make the secondary replication devices available and change their status from being secondary devices to suspended primary devices.
3. If you needed to relax node quorum or make configuration changes, migrate the primary cluster configuration server to site **B**, issue this command:

   ```
   mmchcluster -p nodeB001
   ```

4. If site **C**, the tiebreaker, failed along with site **A**, existing node quorum designations must be relaxed in order to allow the surviving site to fulfill quorum requirements. To relax node quorum, temporarily change the designation of each of the failed quorum nodes to non-quorum nodes:

   ```
   mmchnode --nonquorum -N nodeA001,nodeA002,nodeA003,nodeC
   ```

5. Ensure the source volumes are *not* accessible to the recovery site:
   - Disconnect the cable
   - Define the **nsddevices** user exit file to exclude the source volumes
6. Restart the GPFS daemon on all surviving nodes:

   ```
   mmstartup -N gpfs.siteB
   ```

**Procedure when the Clustered Configuration Repository (CCR) scheme is in use**

For an active-active PPRC-based cluster, follow these steps to restore access to the file system through site **B** after site **A** has experienced a disastrous failure:

1. Stop the GPFS daemon on the surviving nodes as site **B** where the file gpfs.siteB lists all of the nodes at site **B**:

   ```
   mmshutdown -N gpfs.siteB
   ```

2. Perform the appropriate commands to make the secondary replication devices available and change their status from being secondary devices to suspended primary devices.
3. If site **C**, the tiebreaker, failed along with site **A**, existing node quorum designations must be relaxed in order to allow the surviving site to fulfill quorum requirements. To relax node quorum, temporarily change the designation of each of the failed quorum nodes to nonquorum nodes using the `–- force` option:

   ```
   mmchnode --nonquorum -N nodeA001,nodeA002,nodeA003,nodeC --force
   ```

4. Ensure that the source volumes are not accessible to the recovery site:
   - Disconnect the cable.
   - Define the **nsddevices** user exit file to exclude the source volumes.
5. Restart the GPFS daemon on all surviving nodes:

   ```
   mmstartup -N gpfs.siteB
   ```

**Note:**

- Make no further changes to the quorum designations at site B until the failed sites are back on line and the following failback procedure has been completed.
- Do not shut down the current set of nodes on the surviving site B and restart operations on the failed sites A and C. This will result in a non-working cluster.

**Failback procedure**

After the operation of site **A** has been restored, the failback procedure is completed to restore the access to the file system from that location. The following procedure is the same for both configuration schemes (server-based and Clustered Configuration Repository (CCR)). The failback operation is a two-step process:

1. For each of the paired volumes, resynchronize the pairs in the reserve direction with the recovery LUN **diskB** acting as the sources for the production LUN **diskA**. An incremental resynchronization is performed, which identifies the mismatching disk tracks, whose content is then copied from the recovery LUN to the production LUN. Once the data has been copied and the replication is running in the reverse direction this configuration can be maintained until a time is chosen to switch back to site **A**.
2. Shut GPFS down at site **B** and reverse the disk roles (the original primary disk becomes the primary again), bringing the replication pair to its initial state.
   a. Stop the GPFS daemon on all nodes.
   b. Perform the appropriate actions to switch the replication direction so that **diskA** is now the source and **diskB** is the target.
   c. If during failover you migrated the primary cluster configuration server to a node in site **B**:
      1) Migrate the primary cluster configuration server back to site **A**:
         ```
         mmchcluster -p nodeA001
         ```
      2) Restore the initial quorum assignments:
         ```
         mmchnode --quorum -N nodeA001,nodeA002,nodeA003,nodeC
         ```
      3) Ensure that all nodes have the latest copy of the **mmsdrfs** file:
         ```
         mmchcluster -p LATEST
         ```
   d. Ensure the source volumes *are* accessible to the recovery site:
      - Reconnect the cable
      - Edit the **nsddevices** user exit file to *include* the source volumes
   e. Start the GPFS daemon on all nodes:
      ```
      mmstartup -a
      ```
   f. Mount the file system on all the nodes at sites **A** and **B**.

## An active-passive GPFS cluster

In an active-passive environment, two GPFS clusters are set up in two geographically distinct locations (the production and the recovery sites). These clusters are referred to as peer GPFS clusters.

A GPFS file system is defined over a set of disk volumes located at the production site and these disks are mirrored using storage replication to a secondary set of volumes located at the recovery site. During normal operation, only the nodes in the production GPFS cluster mount and access the GPFS file system at any given time, which is the primary difference between a configuration of this type and the active-active model.

In the event of a catastrophe in the production cluster, the storage replication target devices are made available to be used by the nodes in the recovery site.

The secondary replica is then mounted on nodes in the recovery cluster as a regular GPFS file system, thus allowing the processing of data to resume at the recovery site. At a latter point, after restoring the physical operation of the production site, we execute the failback procedure to resynchronize the content of the replicated volume pairs between the two clusters and re-enable access to the file system in the production environment.

The high-level organization of synchronous active-passive storage replication based GPFS cluster is shown in Figure 13.

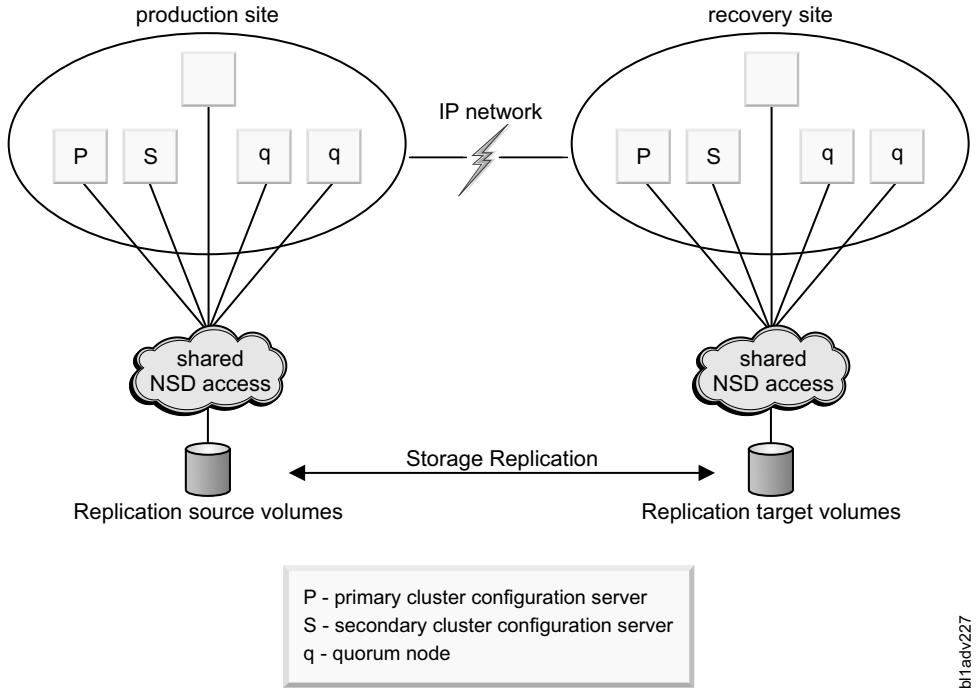



*Figure 13. A synchronous active-passive storage replication-based GPFS configuration without a tiebreaker site*

**Setting up an active-passive GPFS configuration:**

This example demonstrates how to configure an active-passive GPFS cluster.

To establish an active-passive storage replication GPFS cluster as shown in Figure 13, consider the configuration:

**Production site**

Consists of:

- Nodes – **nodeP001**, **nodeP002**, **nodeP003**, **nodeP004**, **nodeP005**
- Storage subsystems – Enterprise Storage Server® **P**
- LUN ids and disk volume names – **lunP1** (**hdisk11**), **lunP2** (**hdisk12**), **lunP3** (**hdisk13**), **lunP4** (**hdisk14**)

**Recovery site**

Consists of:

- Nodes – **nodeR001**, **nodeR002**, **nodeR003**, **nodeR004**, **nodeR005**
- Storage subsystems – Storage System **R**
- LUN ids and disk volume names – **lunR1** (**hdisk11**), **lunR2** (**hdisk12**), **lunR3** (**hdisk13**), **lunR4** (**hdisk14**)

All disks are SAN-attached and directly accessible from all local nodes.

1. Establish synchronous PPRC volume pairs using the **copy entire volume** option:

```
lunP1-lunR1 (source-target)
lunP2-lunR2 (source-target)
lunP3-lunR3 (source-target)
lunP4-lunR4 (source-target)
```

.

2. Create the recovery cluster selecting **nodeR001** as the primary cluster data server node, **nodeR002** as the secondary cluster data server nodes, and the nodes in the cluster contained in the file **NodeDescFileR**. The **NodeDescFileR** file contains the node descriptors:

```
nodeR001:quourum-manager
nodeR002:quourum-manager
nodeR003:quourum-manager
nodeR004:quourum-manager
nodeR005
```

   Issue this command:

```
mmcrcluster –N NodeDescFileR –p nodeR001 –s nodeR002
```

3. Create the GPFS production cluster selecting **nodeP001** as the primary cluster data server node, **nodeP002** as the secondary cluster data server node, and the nodes in the cluster contained in the file **NodeDescFileP**. The **NodeDescFileP** file contains the node descriptors:

```
nodeP001:quourum-manager
nodeP002:quourum-manager
nodeP003:quourum-manager
nodeP004:quourum-manager
nodeP005
```

   Issue this command:

```
mmcrcluster –N NodeDescFileP –p nodeP001 –s nodeP002
```

4. At all times the peer clusters must see a consistent image of the mirrored file system's configuration state contained in the **mmsdrfs** file. After the initial creation of the file system, all subsequent updates to the local configuration data must be propagated and imported into the peer cluster. Execute the **mmfsctl syncFSconfig** command to resynchronize the configuration state between the peer clusters after each of these actions in the primary GPFS cluster:
   - Addition of disks through the **mmadddisk** command
   - Removal of disks through the **mmdeldisk** command
   - Replacement of disks through the **mmrpldisk** command
   - Modifications to disk attributes through the **mmchdisk** command
   - Changes to the file system's mount point through the **mmchfs -T** command

   To automate the propagation of the configuration state to the recovery cluster, activate and use the **syncFSconfig** user exit. Follow the instructions in the prolog of **/usr/lpp/mmfs/samples/syncfsconfig.sample**.
5. From a node in the production cluster, start the GPFS daemon on all nodes:

```
mmstartup -a
```

6. Create the NSDs at the production site. The disk descriptors contained in the file **DiskDescFileP** are:

```
/dev/hdisk11:nodeP001:nodeP002:dataAndMetadata:-1
/dev/hdisk12:nodeP001:nodeP002:dataAndMetadata:-1
/dev/hdisk13:nodeP001:nodeP002:dataAndMetadata:-1
/dev/hdisk14:nodeP001:nodeP002:dataAndMetadata:-1
```

   Issue this command:

```
mmcrnsd –F DiskDescFileP
```

7. Create the GPFS file system and mount it on all nodes at the production site:

```
mmcrfs /gpfs/fs0 fs0 -F DiskDescFileP
```

**Failover to the recovery site and subsequent failback for an active-passive configuration:**

For an active-passive storage replication based cluster, complete these steps to fail over production to the recovery site.

**Procedure when the server-based configuration scheme is in use**

1. If the GPFS daemon is not already stopped on all surviving nodes in the production cluster, from a node in the production cluster issue:

   ```
   mmshutdown –a
   ```

2. Perform the appropriate commands to make the secondary replication devices available and change their status from being secondary devices to suspended primary devices.
3. From a node in the recovery cluster start GPFS:

   ```
   mmstartup –a
   ```

4. Mount the file system on all nodes in the recovery cluster.

**Procedure when the Clustered Configuration Repository (CCR) scheme is in use**

1. Stop the GPFS daemon on the surviving nodes as site **B** where the file `gpfs.siteB` lists all of the nodes at site **B**:

   ```
   mmshutdown -N gpfs.siteB
   ```

2. Perform the appropriate commands to make the secondary replication devices available and change their status from being secondary devices to suspended primary devices.
3. If site **C**, the tiebreaker, failed along with site **A**, existing node quorum designations must be relaxed in order to allow the surviving site to fulfill quorum requirements. To relax node quorum, temporarily change the designation of each of the failed quorum nodes to nonquorum nodes using the `–- force` option:

   ```
   mmchnode --nonquorum -N nodeA001, nodeA002, nodeA003, nodeC --force
   ```

4. Ensure that the source volumes are *not* accessible to the recovery site:
   - Disconnect the cable
   - Define the **nsddevices** user exit file to exclude the source volumes
5. Restart the GPFS daemon on all surviving nodes:

   ```
   mmstartup -N gpfs.siteB
   ```

**Note:** Make no further changes to the quorum designations at site B until the failed sites are back on line and the following failback procedure has been completed. Do not shut down the current set of nodes on the surviving site B and restart operations on the failed sites A and C. This will result in a non-working cluster.

**Failback procedure**

After the physical operation of the production site has been restored, complete the failback procedure to transfer the file system activity back to the production GPFS cluster. The following procedure is the same for both configuration schemes (server-based and Clustered Configuration Repository (CCR)) The failback operation is a two-step process:

1. For each of the paired volumes, resynchronize the pairs in the reserve direction with the recovery LUN **lunRx** acting as the sources for the production LUN **lunPx**. An incremental resynchronization will be performed which identifies the mismatching disk tracks, whose content is then copied from the recovery LUN to the production LUN. Once the data has been copied and the replication is running in the reverse direction this configuration can be maintained until a time is chosen to switch back to site **P**.
2. If the state of the system configuration has changed, update the GPFS configuration data in the production cluster to propagate the changes made while in failover mode. From a node at the recovery site, issue:

   ```
   mmfsctl all syncFSconfig –n gpfs.sitePnodes
   ```

3. Stop GPFS on all nodes in the recovery cluster and reverse the disk roles so the original primary disks become the primaries again:
   a. From a node in the recovery cluster, stop the GPFS daemon on all nodes in the recovery cluster:

```
mmshutdown –a
```

b. Perform the appropriate actions to switch the replication direction so that **diskA** is now the source and **diskB** is the target.

c. From a node in the production cluster, start GPFS:

```
mmstartup –a
```

d. From a node in the production cluster, mount the file system on all nodes in the production cluster.

# Point In Time Copy of IBM Spectrum Scale data

Most storage systems provides functionality to make a point-in-time copy of data as an online backup mechanism. This function provides an instantaneous copy of the original data on the target disk, while the actual copy of data takes place asynchronously and is fully transparent to the user.

Several uses of the FlashCopy replica after its initial creation can be considered. For example, if your primary operating environment suffers a permanent loss or a corruption of data, you may choose to flash the target disks back onto the originals to quickly restore access to a copy of the file system as seen at the time of the previous snapshot. Before restoring the file system from a FlashCopy, please make sure to suspend the activity of the GPFS client processes and unmount the file system on all GPFS nodes. FlashCopies also can be used to create a copy of data for disaster recovery testing and in this case are often taken from the secondary devices of a replication pair.

When a FlashCopy disk is first created, the subsystem establishes a control bitmap that is subsequently used to track the changes between the source and the target disks. When processing read I/O requests sent to the target disk, this bitmap is consulted to determine whether the request can be satisfied using the target's copy of the requested block. If the track containing the requested data has not yet been copied, the source disk is instead accessed and its copy of the data is used to satisfy the request.

To prevent the appearance of out-of-order updates, it is important to consider data consistency when using FlashCopy. When taking the FlashCopy image all disk volumes that make up the file system must be copied so that they reflect the same logical point in time. Two methods may be used to provide for data consistency in the FlashCopy image of your GPFS file system. Both techniques guarantee the consistency of the FlashCopy image by the means of temporary suspension of I/O, but either can be seen as the preferred method depending on your specific requirements and the nature of your GPFS client application.

FlashCopy provides for the availability of the file system's on-disk content in another GPFS cluster. But in order to make the file system known and accessible, you must issue the **mmfsctl syncFSConfig** command to:

- Import the state of the file system's configuration from the primary location.
- Propagate all relevant changes to the configuration in the primary cluster to its peer to prevent the risks of discrepancy between the peer's **mmsdrfs** file and the content of the file system descriptor found in the snapshot.

  It is suggested you generate a new FlashCopy replica immediately after every administrative change to the state of the file system. This eliminates the risk of a discrepancy between the GPFS configuration data contained in the **mmsdrfs** file and the on-disk content of the replica.

## Using consistency groups for Point in Time Copy

This topic provides a description about using consistency groups for point in time copy mechanism in IBM Spectrum Scale.

The use of FlashCopy consistency groups provides for the proper ordering of updates, but this method does not by itself suffice to guarantee the atomicity of updates as seen from the point of view of the user application. If the application process is actively writing data to GPFS, the on-disk content of the file system may, at any point in time, contain some number of incomplete data record updates and possibly

some number of in-progress updates to the GPFS metadata. These appear as partial updates in the FlashCopy image of the file system, which must be dealt before enabling the image for normal file system use. The use of metadata logging techniques enables GPFS to detect and recover from these partial updates to the file system's metadata. However, ensuring the atomicity of updates to the actual data remains the responsibility of the user application. Consequently, the use of FlashCopy consistency groups is suitable only for applications that implement proper mechanisms for the recovery from incomplete updates to their data.

The FlashCopy consistency group mechanism is used to freeze the source disk volumes at the logical instant at which their logical image appears on the target disk volumes. The appropriate storage system documentation should be consulted to determine how to invoke the Point in Time Copy with the consistency group option:

Assuming a configuration with:

- Storage subsystems – 1
- LUN ids and disk volume names – **lunS1** (**hdisk11**), **lunS2** (**hdisk12**), **lunT1**, **lunT2**

  **lunS1** and **lunS2** are the FlashCopy source volumes. These disks are SAN-connected and appear on the GPFS nodes as **hdisk11** and **hdisk12**, respectively. A single GPFS file system **fs0** has been defined over these two disks. **lunT1** and **lunT2** are the FlashCopy target volumes. None of the GPFS nodes have direct connectivity to these disks.

To generate a FlashCopy image using a consistency group, do the following step:

Run the **establish FlashCopy pair** task with the **freeze FlashCopy consistency group** option. Create the volume pairs:

```
lunS1 – lunT1  (source-target)
lunS2 – lunT2  (source-target)
```

## Using file-system-level suspension for Point in Time Copy

The use of file-system-level suspension through the **mmfsctl** command prevents incomplete updates in the FlashCopy image and is the suggested method for protecting the integrity of your FlashCopy images. Issuing the **mmfsctl** command leaves the on-disk copy of the file system in a fully consistent state, ready to be flashed and copied onto a set of backup disks. The command instructs GPFS to flush the data buffers on all nodes, write the cached metadata structures to disk, and suspend the execution of all subsequent I/O requests.

1. To initiate file-system-level suspension, issue the **mmfsctl suspend** command.
2. To resume normal file system I/O, issue the **mmfsctl resume** command.

Assuming a configuration with:

- Storage subsystems – ESS 1; logical subsystem LSS 1
- LUN ids and disk volume names – **lunS1** (**hdisk11**), **lunS2** (**hdisk12**), **lunT1**, **lunT2**

  **lunS1** and **lunS2** are the FlashCopy source volumes. These disks are SAN-connected and appear on the GPFS nodes as **hdisk11** and **hdisk12**, respectively. A single GPFS file system **fs0** has been defined over these two disks.

  **lunT1** and **lunT2** are the FlashCopy target volumes. None of the GPFS nodes have direct connectivity to these disks.

To generate a FlashCopy image using file-system-level suspension:

1. From any node in the GPFS cluster, suspend all file system activity and flush the GPFS buffers on all nodes:

   ```
   mmfsctl fs0 suspend
   ```

2. Run the **establish FlashCopy pair** task to create the following volume pairs:

   ```
   lunS1 – lunT1  (source-target)
   lunS2 – lunT2  (source-target)
   ```

3. From any node in the GPFS cluster, resume the file system activity:

```
mmfsctl fs0 resume
```

# Chapter 27. Implementing a clustered NFS environment on Linux

In addition to the traditional exporting of GPFS file systems using the Network File System (NFS) protocol, GPFS allows you to configure a subset of the nodes in the cluster to provide a highly-available solution for exporting GPFS file systems using NFS.

**Note:** This feature is available with IBM Spectrum Scale Standard Edition or higher.

The participating nodes are designated as Cluster NFS (CNFS) member nodes and the entire setup is frequently referred to as CNFS or a CNFS cluster.

In this solution, all CNFS nodes export the same file systems to the NFS clients. When one of the CNFS nodes fails, the NFS serving load moves from the failing node to another node in the CNFS cluster. Failover is done using recovery groups to help choose the preferred node for takeover. For the NFS client node to experience a seamless failover, hard mounts must be used. The use of soft mounts will likely result in stale NFS file handle conditions when a server experiences a problem, even though CNFS failover will still be done.

Currently, CNFS is supported only in the Linux environment. For an up-to-date list of supported operating systems, specific distributions, and other dependencies, refer to the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

## NFS monitoring

Every node in the CNFS cluster runs a separate GPFS utility that monitors GPFS, NFS, and networking components on the node. Upon failure detection and based on your configuration, the monitoring utility might invoke a failover.

While an NFS server is in a grace period, the NFS monitor sets the server's NFS state to "Degraded". For information about CES states, see the topic *CES monitoring and troubleshooting* in the *IBM Spectrum Scale: Problem Determination Guide*.

## NFS failover

As part of GPFS recovery, the CNFS cluster failover mechanism is invoked. It transfers the NFS serving load that was served by the failing node to another node in the CNFS cluster. Failover is done using recovery groups to help choose the preferred node for takeover.

The failover mechanism is based on IP address failover. The CNFS IP address is moved from the failing node to a healthy node in the CNFS cluster. In addition, it guarantees NFS lock (NLM) recovery.

Failover processing may involve rebooting of the problem node. To minimize the effects of the reboot, it is recommended that the CNFS nodes be dedicated to that purpose and are not used to run other critical processes. CNFS node rebooting should not be disabled or the failover reliability will be severely impacted.

## NFS locking and load balancing

CNFS supports a failover of all of the node's load together (all of its NFS IP addresses) as one unit to another node. However, if no locks are outstanding, individual IP addresses can be moved to other nodes for load balancing purposes.

CNFS is dependent on DNS for any automated load balancing of NFS clients among the NFS cluster nodes. Using the round-robin algorithm is highly recommended.

## CNFS network setup

In addition to one set of IP addresses for the GPFS cluster, a separate set of one or more IP addresses is required for CNFS serving.

The GPFS cluster can be defined over an IPv4 or IPv6 network. The IP addresses specified for CNFS can also be IPv4 or IPv6. The GPFS cluster and CNFS are not required to be on the same version of IP, but IPv6 must be enabled on GPFS to support IPv6 on CNFS.

## CNFS setup

You can set up a clustered NFS environment within a GPFS cluster.

To do this, follow these steps:

1. Designate a separate directory for the CNFS shared files:

   **mmchconfig cnfsSharedRoot=***directory*

   where:

   **cnfsSharedRoot=***directory*
   Is the path name to a GPFS directory, preferably on a small separate file system that is not exported by NFS. The GPFS file system that contains the directory must be configured to be mounted automatically upon GPFS start on each of the CNFS nodes (**-A yes** option on the **mmchfs** command). **cnfsSharedRoot** is a mandatory parameter and must be defined first.
2. Add all GPFS file systems that need to be exported to **/etc/exports**. For NSF export considerations, see the topic *Exporting a GPFS file system using NFS* in the *IBM Spectrum Scale: Administration Guide*.
3. If the shared directory from step 1 is in an exported file system, restrict access to that directory.
4. Use the **mmchnode** command to add nodes to the CNFS cluster:

   **mmchnode --cnfs-interface=***ip_address_list* **-N** *node*

   where:

   *ip_address_list*
   Is a comma-separated list of host names or IP addresses to be used for GPFS cluster NFS serving.

   *node*
   Identifies a GPFS node to be added to the CNFS cluster.

   For more information, see the topic *mmchnode command* in the *IBM Spectrum Scale: Command and Programming Reference*.
5. Use the **mmchconfig** command to configure the optional CNFS parameters.

   **cnfsMountdPort=***mountd_port*
   Specifies the port number to be used for the **rpc.mountd** daemon.

   For CNFS to work correctly with the automounter (AMD), the **rpc.mountd** daemon on the different nodes must be bound to the same port.

   **cnfsNFSDprocs=***nfsd_procs*
   Specifies the number of **nfsd** kernel threads. The default is 32.

   **cnfsVersions=***nfs_versions*
   Specifies a comma-separated list of protocol versions that CNFS should start and monitor. The default is **3,4**. If you are not using NFS v3 and NFS v4, specify this parameter with the appropriate values for your configuration.

**Note:** If you are not using NFS v3 and NFS v4, and you do not explicitly specify **cnfsVersions** with the protocol versions on your system, the following message will continually appear in the **mmfs.log**:

```
Found NFS version mismatch between CNFS and current running config, check the OS config files.
```

6. If multiple failover groups are desired, assign a group ID to each NFS node:

   **mmchnode --cnfs-groupid=***groupid* **-N** *node*

   To assign NFS nodes to different groups, use a group ID that is in a different range of ten. For example, a node with group ID 2*n* will fail over only to nodes in the same range of ten (which means any node with group ID 20 to 29). Failover in the same group will first look for one of the nodes with the same group ID. If none are found, any node in the group range starting at *n*0 to *n*9 is selected.

## CNFS administration

There are some common CNFS administration tasks in this topic along with a sample configuration.

To query the current CNFS configuration, enter:

**mmlscluster --cnfs**

To temporarily disable CNFS on one or more nodes, enter:

**mmchnode --cnfs-disable -N** *NodeList*

**Note:** This operation affects only the high-availability aspects of the CNFS functionality. Normal NFS exporting of the data from the node is not affected. All currently defined CNFS IP addresses remain unchanged. There will be no automatic failover from or to this node in case of a failure. If failover is desired, GPFS should be shut down on the affected node prior to issuing the **mmchnode** command.

To re-enable previously-disabled CNFS member nodes, enter:

**mmchnode --cnfs-enable -N** *NodeList*

**Note:** If the GPFS daemon is running on a node on which CNFS is being re-enabled, the node will try to activate its CNFS IP address. If the IP address was currently on some other CNFS-enabled node, that activation would include a takeover.

To permanently remove nodes from the CNFS cluster, enter:

**mmchnode --cnfs-interface=DELETE -N** *NodeList*

**Note:** This operation affects only the high-availability aspects of the CNFS functionality. Normal NFS exporting of the data from the node is not affected. All currently defined CNFS IP addresses remain unchanged. There will be no automatic failover from or to this node in case of a failure. If failover is desired, GPFS should be shut down on the affected node prior to issuing the **mmchnode** command.

### A sample CNFS configuration

Here is a CNFS configuration example, which assumes the following:

- Your GPFS cluster contains three nodes: `fin18`, `fin19`, and `fin20`
- The host names for NFS serving are: `fin18nfs`, `fin19nfs`, and `fin20nfs`

To define a CNFS cluster made up of these nodes, follow these steps:

1. Add the desired GPFS file systems to **/etc/exports** on each of the nodes.
2. Create a directory called **ha** in one of the GPFS file systems by entering:

   ```
   mkdir /gpfs/fs1/ha
   ```

3. Create a temporary file called **/tmp/hanfs-list**, which contains the following lines:

```
fin18 --cnfs-interface=fin18nfs
fin19 --cnfs-interface=fin19nfs
fin20 --cnfs-interface=fin20nfs
```

4. Set the CNFS shared directory by entering:

   ```
   mmchconfig cnfsSharedRoot=/gpfs/fs1/ha
   ```

5. Create the CNFS cluster with the **mmchnode** command, by entering:

   ```
   mmchnode -S /tmp/hanfs-list
   ```

6. Access the exported GPFS file systems over NFS. If one or more GPFS nodes fail, the NFS clients should continue uninterrupted.

# Chapter 28. Implementing Cluster Export Services

Cluster Export Services (CES) provides highly available file and object services to a GPFS cluster by using Network File System (NFS), Object, or Server Message Block (SMB) protocols.

**Note:** This feature is available with IBM Spectrum Scale Standard Edition or higher.

CES is an alternate approach to using a clustered Network File System (NFS) to export GPFS file systems. For more information about CES and protocol configuration, see Chapter 2, "Configuring the CES and protocol configuration," on page 23.

## CES features

To successfully use Cluster Export Services (CES), you must consider function prerequisites, setup and configuration, failover/failback policies, and other management and administration requirements.

## CES cluster setup

You can set up a Cluster Export Services (CES) environment within a GPFS cluster.

The CES shared root (cesSharedRoot) directory is needed for storing CES shared configuration data, protocol recovery, and some other protocol-specific purposes. It is part of the Cluster Export Configuration and is shared between the protocols. Every CES node requires access to the path that is configured as shared root.

| **Note:** CES clusters are not supported on remote file systems (cross cluster mounts).

To update the CES shared root directory, you must shut down the cluster, set the CES shared root directory, and start the cluster again:

```
mmshutdown -a
mmchconfig cesSharedRoot=shared_root_path
mmstartup -a
```

The recommendation for CES shared root directory is a dedicated file system. It can also reside in an existing GPFS file system. In any case, the CES shared root directory must be on GPFS and must be available when it is configured through the **mmchconfig** command.

To enable protocol nodes, the CES shared root directory must be defined. To enable protocol nodes, use the following command:

```
mmchnode --ces-enable -N Node1[,Node2...]
```

To disable a CES node, use the following command:

```
mmchnode --ces-disable -N Node1[,Node2...]
```

### Preparing to perform service actions on the CES shared root directory file system

The CES shared root directory file system must be kept available for protocols operation to function. If a service action is to be performed on the CES shared root directory file system, perform the steps that follow.

Commands such as **mmshutdown, mmstartup and mmmount**, can be passed in the `cesnodes` node class parameter to ensure operation on all protocol nodes.

The following steps are used to perform service actions on the CES shared root filesystem :

1. Inform users of the impact to protocols. Quiesce protocol related I/O and mounts if possible. Quiesce cluster functions in progress on protocol nodes such as recalls, migrations, AFM, backup, and any policies that may be in use on the protocol nodes; or transition these cluster functions to other nodes. Finally, verify that file system quorum can be achieved by the remaining cluster nodes.
2. Shut down GPFS on all protocol nodes:

   ```
   mmshutdown -N cesnodes
   ```

   **Note:** Only protocol nodes need to be shut down for service of the CES shared root directory file system. However, other nodes may need to unmount the file system, depending on what service is being performed.

Protocol nodes are now ready for service actions to be performed on the CES shared root directory or the nodes themselves. To recover from a service action:

1. Start up GPFS on all protocol nodes:

   ```
   mmstartup -N cesnodes
   ```
2. Make sure that the CES shared root directory file system is mounted on all protocol nodes:

   ```
   mmmount cesSharedRoot -N cesnodes
   ```
3. Verify that all protocol services have been started:

   ```
   mmces service list -a
   ```

## CES network configuration

Cluster Export Services (CES) IP addresses are used to export data via the NFS, SMB, and Object protocols. File and Object clients use the public IPs to access data on GPFS file systems.

CES IP addresses have the following characteristics:

- Shared between all CES protocols
- Organized in an *address pool* (there can be fewer or more CES addresses than nodes)
- Hosted on the CES nodes (there can be CES nodes without CES addresses)
- Can move between CES nodes (triggered manually via the command or as part of a CES failover)
- Must not be used for GPFS communication at the same time

CES IP addresses have these restrictions:

- The network on CES nodes must be configured so that all CES IPs can run on any CES node. Typically this configuration requires that all CES nodes have at least one NIC interface or VLAN-compatible interface with each CES IP network address.
- CES IPs are created as aliases on each CES node. Do not include the primary address of an adapter in the CES IP address pool.
- CES IPs must be resolvable by DNS or `/etc/hosts`.
- CES does not manage the subnet or netmask configuration; it is the user's task.

To add CES IP addresses to the address pool, use the **mmces** command:

```
mmces address add --ces-ip Address[,Address...]
```

By default, addresses are distributed among the CES nodes, but a new address can be assigned to a particular node:

```
mmces address add --ces-ip Address[,Address...] --ces-node Node
```

After a CES IP address is added to the address pool, you can manually move the address to a particular node:

```
mmces address move --ces-ip Address[,Address...] --ces-node Node
```

You can remove a CES IP address from the address pool with the **mmces** command:

```
mmces address remove --ces-ip Address[,Address...]
```

Removing an address while there are clients connected causes the clients to lose those connections. Any reconnection to the removed IP results in a failure. If DNS is used to map a name entry to one or more IP addresses, update the DNS to ensure that a client is not presented an address that was already removed from the pool. This process might also include invalidation of any DNS caches.

### CES addresses are virtual IP addresses

The CES addresses that are assigned to the CES nodes are implemented as IP aliases. Each network adapter that hosts CES addresses must already be configured (with different non-CES IPs) in `/etc/sysconfig`. CES uses the netmask to figure out which interfaces to use. For example, if `eth1` is 10.1.1.1 and `eth2` is 9.1.1.1, then the CES IP 10.1.1.100 maps to `eth1` and the CES IP 9.1.1.100 maps to `eth2`.

Virtual IP addresses include the following advantages:

- The node does not need to wait for the switch to accept the link when an IP address is failed back. Since the address is an alias, the interface on which it resides is already up.
- IP address failover is much faster.
- Administration is simplified by providing a clear distinction between the *system IP* and the *CES IP*. For example, you have a two-node cluster. One of the nodes in the two-node cluster has a problem that induces failover and someone logs in to the suspected node to reboot it. The surviving node might get rebooted by accident if the system address was migrated to the surviving node.

### How to use an alias

To use an alias address for CES, you need to provide a static IP address that is not already defined as an alias in the `/etc/sysconfig/network-scripts` directory.

Before you enable the node as a CES node, configure the network adapters for each subnet that are represented in the CES address pool:

1. Define a static IP address for the device:

   ```
   /etc/sysconfig/network-scripts/ifcfg-eth0
   DEVICE=eth1
   BOOTPROTO=none
   IPADDR=10.1.1.10
   NETMASK=255.255.255.0
   ONBOOT=yes
   GATEWAY=10.1.1.1
   TYPE=Ethernet
   ```

2. Ensure that there are no aliases that are defined in the network-scripts directory for this interface:

   ```
   # ls -l /etc/sysconfig/network-scripts/ifcfg-eth1:*
   ls: /etc/sysconfig/network-scripts/ifcfg-eth1:*: No such file or directory
   ```

After the node is enabled as a CES node, no further action is required. CES addresses are added as aliases to the already configured adapters.

## CES address failover and distribution policies

When a Cluster Export Services (CES) node leaves the GPFS cluster, any CES IP addresses that are assigned to that node are moved to CES nodes still within the cluster. Additionally, certain error conditions and administrative operations can cause a node to release its addresses to be reassigned to other nodes.

As CES nodes enter and leave the GPFS cluster, the addresses are distributed among the nodes according to the address distribution policy that is selected. In addition, you can disable automatic address distribution to allow the user to manually maintain the address-to-node assignments.

The address distribution policy is set with the **mmces** command:

```
mmces address policy [even-coverage | balanced-load | node-affinity | none]
```

The following list describes each type of address distribution policy:

**even-coverage**
: Distributes the addresses among the available nodes. The even-coverage policy is the default address distribution policy.

**balanced-load**
: Distributes the addresses to approach an optimized load distribution. The load (network and CPU) on all the nodes are monitored. Addresses are moved based on given policies for optimized load throughout the cluster.

**node-affinity**
: Attempts to keep an address on the node to which the user manually assigned it. If the **mmces address add** command is used with the `--node` option, the address is marked as being associated with that node. Similarly, if an address is moved with the **mmces address move** command, the address is marked as being associated with the destination node. Any automatic movement, such as reassigning a down node's addresses, does not change this association. Addresses that are enabled with no 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 if possible.

Automatic address distribution is performed in the background in a way as to not disrupt the protocol servers more than necessary. If you want immediate redistribution of the addresses, use the **mmces** command to force an immediate rebalance:

```
mmces address move --rebalance
```

You can further control the assignment of CES addresses by placing nodes or addresses in CES groups. For more information, see the topic “Configuring CES protocol service IP addresses” on page 24.

## CES protocol management

Cluster Export Services (CES) protocols (NFS, SMB, and Object) are enabled or disabled with the **mmces** command.

Command examples:

```
mmces service enable [NFS | OBJ | SMB]
mmces service disable [NFS | OBJ | SMB]
```

After a protocol is enabled, the protocol is started on all CES nodes.

When a protocol is disabled, the protocol is stopped on all CES nodes and all protocol-specific configuration data is removed.

## CES management and administration

Cluster Export Services (CES) nodes can be suspended for maintenance reasons with the **mmces** command.

For example:

```
mmces node suspend [-N Node[,Node...]
```

When a node is suspended:

- The GPFS state of the node is unaffected. It remains an active member of the cluster.
- All CES IP addresses that are assigned to the node are reassigned to other nodes. Assignment of addresses to a suspend node is not allowed.
- All CES monitoring operations are stopped.
- Servers for the enabled CES protocols continue to run, but can be stopped.
- Suspended protocol servers can be stopped and started on the node with the following **mmces** commands.

  ```
  mmces service stop [NFS | OBJ | SMB] [-N Node[,Node...]]
  mmces service start [NFS | OBJ | SMB] [-N Node[,Node...]]
  ```

- A node or a group of nodes can be suspended and resume normal operation with the following **mmces** commands.

  ```
  mmces node suspend -N node1,node2,node3
  mmces node resume
  ```

  After a node is resumed, monitoring on the node is started and the node is eligible for address assignments.

# CES NFS support

In Cluster Export Services (CES), you must consider monitoring, service and export configuration, failover policies, migration, and client support requirements for Network File System (NFS) support.

## NFS support levels

NFS versions 3 (NFSv3) and 4 (NFSv4.0) are supported.

## NFS monitoring

The NFS servers are monitored to check for proper functions. If a problem is found, the CES addresses of the node are reassigned, and the node state is set to `failed`. When the problem is corrected, the node resumes normal operation.

## NFS service configuration

Configuration options for the NFS service can be set with the **mmnfs configuration** command.

You can use the **mmnfs configuration** command to set and list default settings for NFS such as the port number for the NFS service, the default access mode for exported file systems, the log level, and enable or disable status for delegations. For a list of configurable attributes, see the topic *mmnfs command* in the *IBM Spectrum Scale: Command and Programming Reference*.

Some of the attributes such as the protocol can be overruled for a given export on a per-client base. For example, the default settings might have NFS protocols 3 and 4 enabled, but the export for a client might restrict it to NFS version 4 only.

## NFS export configuration

Exports can be added, removed, or changed with the **mmnfs export** command. Authentication must be set up before you define an export.

Exports can be declared for any directory in the GPFS file system, including a fileset junction. At the time where exports are declared, these folders must exist physically in GPFS. Only folders in the GPFS file system can be exported. No folders that are located only locally on a server node can be exported because they cannot be used in a failover situation.

Export-add and export-remove operations can be applied at run time of the NFS service. The export-change operation does require a restart of the NFS service on all server nodes that is followed by a 60-second grace period to allow connected clients to reclaim their locks and to avoid concurrent lock requests from new clients.

## NFS failover

When a CES node leaves the cluster, the CES addresses assigned to that node are redistributed among the remaining nodes. Remote clients that access the GPFS file system might see a pause in service while the internal state information is passed to the new servers.

## NFS clients

When you work with NFS clients, consider the following points:

- If you mount the same NFS export on one client from two different IBM Spectrum Scale NFS protocol nodes, data corruption might occur.
- Cross-protocol notifications are not supported by clustered NFS. For example, files that are created with NFS are not automatically visible on SMB clients and SMB clients need to refresh the directory listing by performing a manual refresh in Microsoft Windows Explorer. Similarly, files that are created from other methods such as files that are created with FTP or files that are restored from a backup are not automatically visible.
- The NFS protocol version that is used as the default on a client operating system might differ from what you expect. If you are using a client that mounts NFSv3 by default, and you want to mount NFSv4, then you must explicitly specify NFSv4 in the **mount** command. For more information, see the **mount** command for your client operating system.
- A client must mount an NFS export by using the IP address of the GPFS system. If a host name is used, ensure that the name is unique and remains unique.

  If a DNS Round Robin (RR) entry name is used to mount an NFSv3 export, data corruption and data unavailability might occur. The lock manager on the GPFS file system is not *clustered-system-aware*.
- Reusing an export ID might cause unexpected behavior on a running system. When an export is removed, avoid reusing the export ID either for a different export or for reexporting the same file system.

## Coexistence with clustered NFS and migration

CES and clustered NFS cannot exist in the same GPFS cluster.

## Choosing between CNFS or CES

If you want to put highly available NFS services on top of the GPFS file system, you have the choice between clustered NFS (Chapter 27, “Implementing a clustered NFS environment on Linux,” on page 389) and Cluster Export Services (Chapter 28, “Implementing Cluster Export Services,” on page 393).

To help you choose one of these NFS offerings, consider the following points:

**Multiprotocol support**

If you plan to use other protocols (such as SMB or Object) in addition to NFS, CES must be chosen. While CNFS provides support only for NFS, the CES infrastructure adds support also for SMB and Object. With CES, you can start with NFS and add (or remove) other protocols at any time.

**Command support**
While CNFS provides native GPFS command support for creation and management of the CNFS cluster, it lacks commands to manage the NFS service and NFS exports. The CES infrastructure introduces native GPFS commands to manage the CES cluster. Furthermore, there are also commands to manage the supported protocol services and the NFS exports. For example, with CES, you do not need to adapt NFS configuration files individually on the protocol nodes. This work is done by the new GPFS commands that are provided for CES.

**Performance**
CNFS is based on the kernel NFS server while NFS support in CES is based on the Ganesha NFS server operating in user space. Due to the different nature of these NFS I/O stacks, performance depends on system characteristics and NFS workload. Contact your IBM representative to get help with sizing the required number of protocol nodes to support certain workload characteristics and protocol connection limits.

There is no general answer about which of the two NFS servers performs better as this depends on many factors. Tests that are conducted with both NFS I/O stacks over various workloads show that the kernel-based NFS server (CNFS) performs better under metadata-intensive workloads. Typically this testing is with many smaller files and structures. The Ganesha NFS server provides better performance on other data-intensive workloads such as video streaming.

**Note:** CES provides a different interface to obtain performance metrics for NFS. CNFS uses the existing interfaces to obtain NFS metrics from the kernel (such as `nfsstat` or the `/proc` interface). The CES framework provides the **mmperfmon query** command for Ganesha-based NFS statistics. For more information, see the topic *mmperfmon command* in the *IBM Spectrum Scale: Command and Programming Reference*.

**Migration of CNFS to CES**
For information about migrating existing CNFS environments to CES, see "Migration of CNFS clusters to CES clusters" on page 403.

---

# CES SMB support

In GPFS 4.2.1, you can access a GPFS file system with an SMB client using its inherent SMB semantics.

The following features are provided:

## Clustered SMB support

SMB clients can connect to any of the protocol nodes and get access to the shares defined. A clustered registry makes sure that all nodes see the same configuration data. Therefore, clients can connect to any Cluster Export Services (CES) node and see the same data. Moreover, the state of opened files (share modes, open modes, access masks, locks, and so on) is also shared among the CES nodes so that data integrity is maintained. On failures, clients can reconnect transparently to another cluster node as the IP addresses of failing nodes are transferred to another cluster node.

The supported protocol levels are SMB2 and the base functions of SMB3 (dialect negotiation, secure negotiation, encryption of data on the wire).

## Export management command

With the **mmsmb** command, IBM Spectrum Scale provides a comprehensive entry point to manage all SMB-related configuration tasks like creating, changing, and deleting SMB shares.

### SMB monitoring

The monitoring framework detects issue with the SMB services and triggers failover in case of an unrecoverable error.

### Integrated installation

The SMB services are installed by the integrated installer together with the CES framework and the other protocols NFS and Object.

### SMB performance metrics

The SMB services provide two sets of performance metrics that are collected by the performance monitor framework. Both current and historic data (with lower granularity) can be retrieved. The two sets of metrics are global SMB metrics (like number of connects and disconnects) and metrics on for each SMB request (number, time, throughput). The mmperfmon query tool provides access to the most important SMB metrics via predefined queries. Moreover, metrics for the clustered file metadata database CTDB are collected and exposed via the **mmperfmon query** command.

### Cross-protocol integration with NFS

In IBM Spectrum Scale 4.1.1 and later, you can concurrently access the same file data with SMB, NFS, and native POSIX access. You cannot concurrently access the same file data using traditional file protocols.

### Authentication and mapping

The SMB services can be configured to authenticate against the authentication services MS Active Directory and LDAP. Mapping MS security identifiers (SIDs) to the POSIX user and group IDs on the file server can either be done automatically by using the so-called autorid mechanism or external mapping services like RFC 2307 or MS Services for Unix. If none of the offered authentication and mapping schemes matches the environmental requirements, a user-defined configuration can be established.

## CES OBJ support

In Cluster Export Services (CES), you must consider several types of requirements for Object (OBJ) support.

### Openstack support levels

In Release 4.2.1, the Liberty release of OpenStack is used for Swift, Keystone and their dependent packages.

The Swift V1 and Keystone V2 and V3 APIs are also supported.

### Object monitoring

The object servers are monitored to ensure that they function properly. If a problem is found, the CES addresses of the node are reassigned, and the node state is set to failed. When the problem is corrected, the node resumes normal operation.

### Object service configuration

The Object service configuration is controlled by the respective Swift and Keystone configuration files. The master versions of these files are stored in the CCR repository, and copies exist in the `/etc/swift` and `/etc/keystone` directories on each protocol node. The files that are stored in those directories should not be directly modified since they are overwritten by the files that are stored in the CCR. To change the

Swift or Keystone configuration, use the **mmobj config change** command to modify the master copy of configuration files stored in CCR. The monitoring framework is notified of the change and propagates the file to the local file system of the CES nodes. For information about the values that can be changed and their associated function, refer to the administration guides for Swift and Keystone.

To change the authentication that is used by the Keystone server, use the **mmuserauth** command to change the authentication repository to AD or LDAP, or to enable SSL communication to the Keystone server.

## Object fileset configuration

A separate fileset must be created for the Object service. This fileset is automatically created in the GPFS file system that is specified during installation. Evaluate the data that is expected to be stored by the Object service to determine the required number of inodes that are needed, which can also be specified during installation.

## Object failover

When a CES node leaves the cluster, the CES addresses that are assigned to that node are redistributed among the remaining nodes. Remote clients that access the Object service might see active connections drop or a pause in service while the while the CES addresses are moved to the new servers. Clients with active connections to the CES addresses that are migrated might have their connections unexpectedly drop. Clients are expected to retry their requests when this happens.

Certain Object-related services can be migrated when a node is taken offline. If the node was hosting the backend database for Keystone or certain Swift services that are designated as singletons (such as the auditor), those services are started on the active node that received the associated CES addresses of the failed node. Normal operation of the Object service resumes after the CES addresses are reassigned and necessary services automatically restarted.

## Object clients

The Object service is based on Swift and Keystone, and externalizes their associated interfaces. Clients should follow the associated specifications for those interfaces. Clients must be able to handle dropped connections or delays during CES node failover. In such situations, clients should retry the request or allow more time for the request to complete.

To connect to an Object service, clients should use a DNS service that returns a single CES address from a pool of addresses or connect to a load balancer that distributes requests among the CES IP addresses. Clients should not use hard-coded CES addresses to connect to Object services. For example, the authentication URL should refer to a DNS host name or a load balancer front end name such as `http://protocols.gpfs.net:35357/v3` rather than a CES address.

## Inode Allocation Overview

Object storage consumes fileset inodes when the unified file and object access layout is used. One inode is used for each file or object, and one inode is used for each directory in the object path. There are no other meta files used.

In the traditional object layout, objects are placed in the following directory path:

*gpfs filesystem root*/*fileset*/o/*virtual device*/objects/*partition*/*hash_suffix*/*hash*/*object*

An example object path is:

```
/ibm/gpfs/objfs/o/z1device111/objects/11247/73a/afbeca778982b05b9dddf4fed88f773a/
1461036399.66296.data
```

Similarly, account and container databases are placed in the following directory paths:

*gpfs filesystem root*/*fileset*/ac/*virtual device*/containers/*partition*/*hash_suffix*/*hash*/*account db*
and

*gpfs filesystem root*/*fileset*/ac/*virtual device*/containers/*partition*/*hash_suffix*/*hash*/*container db*.

An example account path is:

```
/ibm/gpfs/objfs/ac/z1device62/accounts/13700/f60/d61003e46b4945e0bbbfcee341d30f60/
d61003e46b4945e0bbbfcee341d30f60.db
```

An example container path is:

```
/ibm/gpfs/objfs/ac/z1device23/containers/3386/0a9/34ea8d244872a1105b7df2a2e6ede0a9/
34ea8d244872a1105b7df2a2e6ede0a9.db
```

Starting at the bottom of the object path and working upward, each new object that is created requires a new hash directory and a new object file, thereby consuming two inodes. Similarly, for account and container data, each new account and container require a new hash directory and a db file. Also, a db.pending and a lock file is required to serialize access. Therefore, four inodes are consumed for each account and each container at the hash directory level.

If the parent directories do not already exist, they are created, thereby consuming additional inodes. The hash suffix directory is three hexadecimal characters, so there can be a maximum of 0xFFF or 4096 suffix directories per partition. The total number of partitions is specified during initial configuration. For IBM Spectrum Scale, 16384 partitions are allocated to objects and the same number is allocated to accounts and containers.

For each object partition directory, the hashes.pkl file is created to track the contents of the partition subdirectories. Also, there is a .lock file that is created for each partition directory to serialize updates to hashes.pkl. This is a total of three inodes required for each object partition.

There are 128 virtual devices allocated to object data during initial configuration, and the same number is allocated to account and container data. For each virtual device a tmp directory is created to store objects during upload. In the async_pending directory, container update requests that time out are stored until they are processed asynchronously by the object updater service.

The total number of inodes used for object storage in the traditional object layout can be estimated as follows:

```
total required inodes = account & container inodes + object inodes
```

As per this information, there are four inodes per account hash directory and four inodes per container hash directory. In the worst case, there would be one suffix directory, one partition directory, and one virtual device directory for each account and container. Therefore, the maximum inodes for accounts and containers can be estimated as:

```
account and container inodes = (7 * maximum number of accounts) + (7 * maximum number of containers)
```

In a typical object store there are more objects than containers, and more containers than accounts. Therefore, while estimating the required inodes, we estimate the number of inodes required for accounts and containers to be seven times the maximum number of containers. The maximum required inodes can be calculated as shown below:

```
max required inodes = (inodes for objects and hash directory) + (inodes required for hash directories) +
        (inodes required for partition directories and partition metadata) +
        (inodes required for virtual devices) + (inodes required for containers)
max required inodes = (2 x maximum number of objects) + (4096 inodes per partition * 16384 partiions) +
        (16384 partitions * 3) + (128 inodes) + (7 * maximum number of containers)
```

**Important:** As the number of objects grows, the inode requirement is dominated by the number of objects. A safe rule of thumb is to allocate three inodes per expected object when there are 10M to 100M expected objects. For more than 100M, you can allocate closer to 2.5 inodes per object.

**Note:** This applies to a case when all objects as well as account and container data are in the same fileset. While using multiple storage policy filesets or a different fileset for account and container data, the calculations must be adjusted.

# Migration of CNFS clusters to CES clusters

If your system has established clustered Network File System (NFS) clusters, you might consider migrating these clusters to Cluster Export Services (CES) clusters.

## Points to consider before you migrate

Cluster Export Services (CES) protocol nodes have the following dependencies and restrictions:

- CES nodes cannot coexist with CNFS clusters.
- CES NFS does not support the concept of failover groups. If this feature in CNFS is important to you, you might want to reconsider migrating to CES NFS until you no longer need this feature.
- The **spectrumscale** installation toolkit confirms that your cluster does not have an existing CNFS cluster.
- CES nodes use SMB, NFS, and Openstack SWIFT Object services.
- File system ACL permissions need to be in NFSv4 format.
- CES NFS services expects NFSv4 ACL formats.
- CES SMB (Samba) services expects NFSv4 ACL formats
- Existing CNFS exports definitions are not compatible with CES NFS. It is best to script and automate the creation of the equivalent exports by using the **mmnfs export add** command to reduce the amount you need to change in the future.
- CES nodes need authentication that is configured.
- CES nodes do not support IP failover groups (as found in CNFS).
- There is a maximum of 16 protocol nodes in a CES cluster if the SMB protocol is also enabled.
- There is a maximum of 32 protocol nodes in a CES cluster if only NFS is enabled.

Because there is a mutual exclusivity between CNFS and CES nodes, you need to accommodate user and application access outage while CES clusters nodes are installed, configured, set up for authentication, and the NFS exports are re-created. The duration of this process depends on the complexity of the customer environment.

You might want to procure new CES nodes or reuse the existing CNFS nodes. Either way, you cannot use the**spectrumscale** installation toolkit until the CNFS nodes are unconfigured.

If you do not have an opportunity to test or plan the implementation of a CES cluster elsewhere, you might have to deal with the design and implementation considerations and issues during the planned outage period. Usually this process is straightforward and quick. If you have a more complex environment, however, it might take longer than the allotted upgrade window to complete the migration. In this case, it is possible to set up one or two non-CNFS, NFS servers to serve NFS for a short time. During this time, you would move all your CNFS IPs to these nodes as you decommission the CNFS cluster. Then, after you successfully set up your CES nodes, authentication, and corresponding exports,

you can move the IPs from the temporary NFS servers over to the CES nodes.

## Saving CNFS export configuration

You need to make a copy of the exports configuration file `/etc/exports` so that you can use this file as the basis for creating the new exports in CES NFS. CES NFS exports configuration needs to be created by using the **mmnfs export add** command or created in bulk by using the **mmnfs export load** command. When you unconfigure CNFS, you also need to delete the `/etc/exports` file from each of the CNFS nodes.

## Steps to unconfigure CNFS

1. CES nodes can be configured only on Red Hat Enterprise Linux 7.x and SLES 12 nodes. If you are planning to convert your existing CNFS nodes to CES nodes, ensure that they are upgraded to Red Hat Enterprise Linux 7.x or SLES 12 first. It is best to upgrade the nodes first while they are running CNFS so that you can ensure that functions are the same as before you start to change over to CES nodes.
2. Ensure that you stop application and user access to the CNFS exports.
3. Run the **mmchnode** command to dereference / evict a CNFS node from the cluster. This command removes both the node and its associated IP):

   ```
   mmchnode –cnfs-interface=default –N node1Name,node2Name,...
   ```

4. When you remove the last node, CNFS is unconfigured and you see an output similar to this result:

   ```
   [root@esnode3 ~]# mmlscluster --cnfs

   GPFS cluster information
   ========================
     GPFS cluster name:         esvcluster1.esnode1
     GPFS cluster id:           15635445795275488305

   mmlscluster:  CNFS is not defined in this cluster.

   [root@esnode3 ~]#
   ```

5. Consider de-refencing the GPFS variable *cnfsSharedRoot*, although this step is not a requirement.
6. You can now delete the `/etc/exports` file on each of the CNFS nodes. Ensure that you have a backup copy of this file to use as a reference when you create the exports under CES NFS.

## Steps to Configure CES NFS

1. If you have not yet configured the CES nodes for authentication, complete this step before you create the exports. Refer to "CES NFS support" on page 397 for details on configuring authentication for your environment.
2. Ensure that the file systems you want to export access to are configured for the NFSv4 security model. If you are converting an existing file system from another security model to NFSv4 you might need to review the ACL structures of the files and verify that your access will work as expected.
3. If you have special configurations or options set in CNFS server, you might also want to reflect these settings in CES NFS. You need to review the appropriateness of these settings for the new environment. To change the settings, use the following command:

   ```
   mmnfs configuration change
   ```

4. If you have many exports to be converted to CES NFS, use the following command:

   ```
   mmnfs export load ExportCfgFile
   ```

   *ExportCfgFile* contains a listing of all your exports as defined in the format that is used for `/etc/ganesha/gpfs.ganesha.exports.conf`.
5. Alternately, you can manually re-create each export on the CES cluster by using the **mmnfs** command.

   ```
   mmnfs export add Path --client ClientOptions
   ```

6. Before you proceed to configure CES nodes, remove the NFS exports from `/etc/exports` from each of the old CNFS nodes
7. Add the IPs that were previously assigned to CNFS to the address pool to be managed by CNFS by using the following command:

   ```
   mmces address add –-node node1Name –-ces-ip ipAddress
   ```

   See “CES network configuration” on page 394 for details about how to use this command.
8. Take care to ensure that the IP addresses are unique and valid for your subnet.

For more information on creating protocol data exports, see *Fileset considerations for creating protocol data exports* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

## Test access to new exports on CES NFS

Test and verify that you have the same level of access to the NFS exports as you did on CNFS to ensure that your applications and NFS clients can continue without further changes.

# Chapter 29. Identity management on Windows

GPFS allows file sharing among AIX, Linux, and Windows nodes. AIX and Linux rely on 32-bit user and group IDs for file ownership and access control purposes, while Windows uses variable-length security identifiers (SIDs). The difference in the user identity description models presents a challenge to any subsystem that allows for heterogeneous file sharing.

GPFS uses 32-bit ID namespace as the canonical namespace, and Windows SIDs are mapped into this namespace as needed. Two different mapping algorithms are used (depending on system configuration):

- GPFS built-in auto-generated mapping
- User-defined mappings stored in the Microsoft Windows Active Directory using the Microsoft Identity Management for UNIX (IMU) component

## Auto-generated ID mappings

Auto-generated ID mappings are the default. If no explicit mappings are created by the system administrator in the Active Directory using Microsoft Identity Management for UNIX (IMU), all mappings between security identifiers (SIDs) and UNIX IDs will be created automatically using a reserved range in UNIX ID space.

**Note:** If you have a mix of GPFS running on Windows and other Windows clients accessing the integrated SMB server function, the ability to share data between these clients has not been tested or validated. With protocol support, the SMB server may also be configured to automatically generate ID mapping. If you want to ensure that SMB users do not access data (share ID mapping) with Windows users, ensure that the automatic range for SMB server is different from this range. The range of IDs automatically generated for the SMB server can be controlled by **mmuserauth**.

Unless the default reserved ID range overlaps with an ID already in use, no further configuration is needed to use the auto-generated mapping function. If you have a specific file system or subtree that are only accessed by user applications from Windows nodes (even if AIX or Linux nodes are used as NSD servers), auto-generated mappings will be sufficient for all application needs.

The default reserved ID range used by GPFS starts with ID 15,000,000 and covers 15,000,000 IDs. The reserved range should not overlap with any user or group ID in use on any AIX or Linux nodes. To change the starting location or the size of the reserved ID range, use the following GPFS configuration parameters:

**sidAutoMapRangeLength**
Controls the length of the reserved range for Windows SID to UNIX ID mapping.

**sidAutoMapRangeStart**
Specifies the start of the reserved range for Windows SID to UNIX ID mapping.

**Note:** For planning purposes, remember that auto-generated ID mappings are stored permanently with file system metadata. A change in the **sidAutoMapRangeStart** value is only effective for file systems created after the configuration change.

## Installing Windows IMU

The Identity Management for UNIX (IMU) feature is included in Windows Server. This feature needs to be installed on the primary domain controller, as well as on any backup domain controllers. It is not installed by default. There are two components that need to be installed in order for IMU to function correctly.

When Active Directory is running on Windows Server 2008, follow these steps to add the IMU service:

1. Open Server Manager.
2. Under Roles, select **Active Directory Domain Services**.
3. Under Role Services, select **Add Role Services**.
4. Under the Identity Management for UNIX role service, check **Server for Network Information Services**.
5. Click **Next**, then **Install**.
6. Restart the system when the installation completes.

## Configuring ID mappings in IMU

To configure ID mappings in Microsoft Identity Management for UNIX (IMU), follow the steps in this procedure.

1. Open **Active Directory Users and Computers** (accessible under Administrative Tools).
2. Select the **Users** branch in the tree on the left under the branch for your domain to see the list of users and groups in this domain.
3. Double-click on any user or group line to bring up the Properties window. If IMU is set up correctly, there will be a **UNIX Attributes** tab as shown in Figure 14:

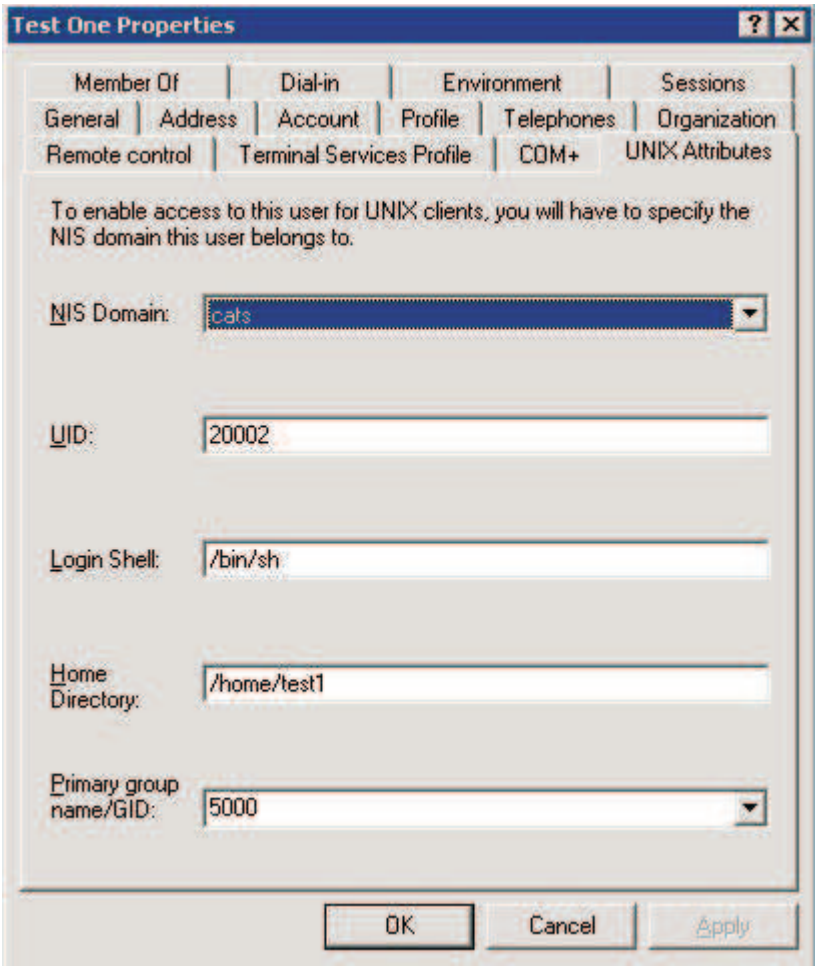



*Figure 14. Properties window*

**Note:** Because the IMU subsystem was originally designed to support integration with the UNIX Network Information Service (NIS), there is an **NIS Domain** field in the Properties window. You do not need to have NIS set up on the UNIX side. For GPFS, the NIS language does not matter.

Update information on the **UNIX Attributes** panel as follows:

1. Under the **NIS Domain** drop-down list, select the name of your Active Directory domain. Selecting **<none>** will remove an existing mapping.
2. Specify a UID in the **UID** field, and for Group objects, specify a GID. This will create a bidirectional mapping between the corresponding SID and a UNIX ID. IMU will disallow the use of the same UID or GID for more than one user or group to ensure that all mappings are unique. In addition to creating mappings for domain users and groups, you can create mappings for certain built-in accounts by going to the **Builtin branch** in the Active Directory Users and Computers panel.
3. Disregard the **Primary group name/GID** field because GPFS does not use it.

It is generally better to configure all ID mappings before mounting a GPFS file system for the first time. Doing that ensures that GPFS only stores properly remapped IDs on disk. However, it is possible to add or delete mappings at any time while GPFS file systems are mounted. GPFS picks up mapping changes dynamically (the code currently checks for mapping changes every 60 seconds), and will start using them at that time.

If an IMU mapping is configured for an ID that is already recorded in some file metadata, you must proceed with caution to avoid user confusion and access disruption. Auto-generated mappings already stored in access control lists (ACLs) on disk will continue to map correctly to Windows SIDs. However, because the SID is now mapped to a different UNIX ID, when you access a file with an ACL containing its auto-generated ID, this access will effectively appear to GPFS as an access by a different user. Depending on the file access permissions, you might not be able to access files that were previously accessible. Rewriting affected ACLs after setting up a new mapping will help replace auto-generated IDs with IMU-mapped IDs, and will restore proper file access for the affected ID (this operation might need to be performed by the system administrator). Examining file ownership and permission information from a UNIX node (for example, using the **mmgetacl** command) is the easiest way to determine whether the ACL for a specified file contains auto-generated or IMU-mapped IDs.

# Chapter 30. Protocols cluster disaster recovery

Protocols cluster disaster recovery (DR) uses the capabilities of Active File Management (AFM) based Async Disaster Recovery (AFM DR) features to provide a solution that allows an IBM Spectrum Scale cluster to fail over to another cluster and fail back, and backup and restore the protocol configuration information in cases where a secondary cluster is not available.

For more information on AFM-based Async DR, see the topic *Introduction to AFM-based Asynchronous Disaster Recovery (AFM DR)* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

**Important:** Our initial feedback from the field suggests that success of a disaster recovery solution depends on administration discipline, including careful design, configuration and testing. Considering this, IBM has decided to disable the Active File Management-based Asynchronous Disaster Recovery feature (AFM DR) by default and require that customers deploying the AFM DR feature first review their deployments with IBM Spectrum Scale development. You should contact IBM Spectrum Scale Support at scale@us.ibm.com to have your use case reviewed. IBM will help optimize your tuning parameters and enable the feature. Please include this message while contacting IBM Support.

These limitations do not apply to base AFM support. These apply only to Async DR available with the IBM Spectrum Scale Advanced Edition V4.2 and V4.1.1.

For more information, see Flash (Alert): IBM Spectrum Scale (GPFS) V4.2 and V4.1.1 AFM Async DR requirement for planning.

Although an overview of the steps that need to be done is provided if performing these operations manually, it is recommended to use the **mmcesdr** command because it automates DR setup, failover, failback, backup, and restore actions. For more information about the **mmcesdr** command, see *mmcesdr command* in *IBM Spectrum Scale: Command and Programming Reference* .

## Protocols cluster disaster recovery limitations and prerequisites

For protocols cluster disaster recovery (DR) in an IBM Spectrum Scale cluster, the prerequisites and limitations are as follows.

Ensure that the following prerequisites are met for the secondary cluster for disaster recovery in an IBM Spectrum Scale with protocols.

- IBM Spectrum Scale is installed and configured.
- Cluster Export Services (CES) are installed and configured, and the shared root file system is defined.
- All protocols that are configured on the primary cluster are also configured on the secondary cluster.
- Authentication on secondary cluster is identical to the authentication on the primary cluster.
- All exports that need to be protected using AFM DR must have the same device and fileset name, and the same fileset link point on the secondary cluster as defined on the primary cluster.
- IBM NFSv3 stack must be configured on home cluster for the AFM DR transport of data.
- No data must be written to exports on secondary cluster while cluster is acting only as a secondary cluster, before a failover.

The following limitations apply for disaster recovery in an IBM Spectrum Scale cluster with protocols.

- Only data contained within independent filesets can be configured for AFM based Async Disaster Recovery (AFM DR). Therefore, all protocol exports that you want to be protected by DR must have the export path equal to the independent fileset link point.

- Nested independent or dependent filesets are not supported.
- Backup and restore of the authentication configuration is not supported.
- On failover and failback or restore, all clients need to disconnect and then reconnect.
- If **--file-config --restore** is specified, perform the follow steps:
  - On failover: file authentication must be removed and then reconfigured on the secondary cluster.
  - On restore: file authentication must be removed and then reconfigured on the primary cluster.
  - On failback: file authentication must be removed and then reconfigured on both primary and secondary clusters.
- Multi-region object deployment is not supported with protocols cluster DR. Therefore, if multi-region object deployment is enabled, object data or configuration information is not protected through protocols cluster DR.
- IBM Spectrum Protect for Space Management and IBM Spectrum Archive migrated data within protocol exports is not supported within protocols cluster DR.
- IBM Spectrum Protect configuration file information is not automatically protected through protocols cluster DR.

## Example setup for protocols disaster recovery

The following example scenario is used to show how to set up disaster recovery functionality for an IBM Spectrum Scale cluster with protocols.

This example consists of three NFS exports, three SMB shares, one object fileset, and two unified file and object access filesets that are also NFS exports. For the SMB and NFS exports, only two of each are independent filesets. This allows an AFM-based Async DR (AFM DR) configuration. For simplification, the filesets are named according to whether or not they were dependent or independent for the SMB and NFS exports. The inclusion of dependent filesets as exports is to show the warnings that are given when an export path is not an independent fileset link point.

**Note:** SMB and NFS exports must be named according to their fileset link point names for them to be captured by the **mmcesdr** command for protocols cluster disaster recovery. For example, if you have a fileset `nfs-smb-combo`, the NFS or the SMB export name must be *GPFS_Path*`/nfs-smb-combo`. If you use a name in the fileset's subdirectory for the SMB or the NFS export (for example: *GPFS_Path*`/nfs-smb-combo/nfs1`), the **mmcesdr** command does not capture that export.

**NFS exports**

- `/gpfs/fs0/nfs-ganesha-dep`
- `/gpfs/fs0/nfs-ganesha1`
- `/gpfs/fs0/nfs-ganesha2`

**SMB shares**

- `/gpfs/fs0/smb1`
- `/gpfs/fs0/smb2`
- `/gpfs/fs0/smb-dep`

**Combination SMB and NFS exports**

- `/gpfs/fs0/combo1`
- `/gpfs/fs0/combo2`

**Object fileset**

- `/gpfs/fs1/object_fileset`

**Unified file and object access filesets**

- `/gpfs/fs1/obj_sofpolicy1`

  This fileset is created by creating a storage policy using the **mmobj policy create sofpolicy1 --enable-file-access** command.

- `/gpfs/fs1/obj_sofpolicy2`

  This fileset is created by creating a storage policy using the **mmobj policy create sofpolicy2 --enable-file-access --enable-compression --compression-schedule "30:02:*:*"** command.

## Setting up gateway nodes to ensure cluster communication during failover

Both the primary and the DR clusters require designating gateway nodes for access to whichever side is acting as the cache. By designating gateway nodes on both clusters, you can ensure that even during failover, cluster communication continues properly.

To handle a possible node failure, you need to specify at least two nodes on each cluster to be gateway nodes. To specify two nodes on the primary cluster as gateway nodes, use the command similar to the following:

```
mmchnode -N Node1,Node2 --gateway
```

Using the example setup mentioned in "Example setup for protocols disaster recovery" on page 412, the command to specify gateway nodes on the primary cluster is as follows:

```
# mmchnode -N zippleback-vm1,zippleback-vm2 --gateway
Tue Apr 28 20:59:01 MST 2015: mmchnode: Processing node zippleback-vm2
Tue Apr 28 20:59:01 MST 2015: mmchnode: Processing node zippleback-vm1
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
# Tue Apr 28 20:59:04 MST 2015: mmcommon pushSdr_async:
mmsdrfs propagation started

Tue Apr 28 20:59:08 MST 2015: mmcommon pushSdr_async:
mmsdrfs propagation completed; mmdsh rc=0
```

Similarly, you need to specify at least two nodes on the DR cluster as gateway nodes. Using the example setup, the command to specify gateway nodes on the DR cluster is as follows:

```
# mmchnode -N windwalker-vm1,windwalker-vm2 --gateway
Tue Apr 28 20:59:49 MST 2015: mmchnode: Processing node windwalker-vm2
Tue Apr 28 20:59:49 MST 2015: mmchnode: Processing node windwalker-vm1
mmchnode: Propagating the cluster configuration data to all
affected nodes. This is an asynchronous process.
# Tue Apr 28 20:59:51 MST 2015: mmcommon pushSdr_async:
mmsdrfs propagation started

Tue Apr 28 20:59:54 MST 2015: mmcommon pushSdr_async:
mmsdrfs propagation completed; mmdsh rc=0
```

## Creating the inband disaster recovery setup

Use the following steps for inband disaster recovery setup in an IBM Spectrum Scale cluster with protocols.

1. On the primary cluster, use the following command to configure independent fileset exports as AFM DR filesets and back up 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 the following:

```
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
```

**Note:** In this command example, there are two exports that are not protected. During the configuration step, any exports that are not protected through AFM DR generate a warning to the standard output of the command.

2. Use the following command to transfer the DR configuration file from the primary cluster to the secondary cluster.

   ```
   scp /root//DR_Config windwalker-vm1:/root/
   ```

   The system displays output similar to the following:

   ```
   root@windwalker-vm1's password:
   DR_Config 100% 1551 1.5KB/s 00:00
   ```

3. On the secondary cluster, use the following command to create the independent filesets that will be a part of the pair of AFM DR filesets associated with those on the primary cluster. In addition to creating filesets, this command also creates the necessary NFS exports.

   ```
   mmcesdr secondary config --input-file-path /root/ --inband
   ```

   The system displays output similar to the following:

   ```
   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.
   ```

4. Ensure that all of the expected AFM DR pairs show as `Active` in the output of the **mmafmctl** command and take corrective action if they do not.

   ```
   # mmafmctl fs0 getstate

   Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
   ------------ -------------- ------------- ------------ ------------ -------------
   nfs-ganesha1 nfs://9.11.102.210/gpfs/fs0/nfs-ganesha1 Active zippleback-vm2 0 4
   nfs-ganesha2 nfs://9.11.102.211/gpfs/fs0/nfs-ganesha2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 4
   combo1 nfs://9.11.102.210/gpfs/fs0/combo1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 7
   combo2 nfs://9.11.102.211/gpfs/fs0/combo2 Active zippleback-vm2 0 66
   smb1 nfs://9.11.102.210/gpfs/fs0/smb1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 65
   smb2 nfs://9.11.102.211/gpfs/fs0/smb2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 4

   # mmafmctl fs1 getstate

   Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
   ------------ -------------- ------------- ------------ ------------ -------------
   object_fileset nfs://9.11.102.211/gpfs/fs1/object_fileset Active zippleback-vm1.tuc.stglabs.ibm.com 0 95671
   obj_sofpolicy1 nfs://9.11.102.211/gpfs/fs1/obj_sofpolicy1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 27
   obj_sofpolicy2 nfs://9.11.102.210/gpfs/fs1/obj_sofpolicy2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 26
   async_dr nfs://9.11.102.210/gpfs/fs1/.async_dr Active zippleback-vm1.tuc.stglabs.ibm.com 0 2751
   ```

   **Note:** A state of `Dirty` is normal when data is actively being transferred from the primary cluster to the secondary cluster.

### Alternate Inband Set up showing the `--allowed-nfs-clients` parameter being used

With the addition of the new optional parameter **--allowed-nfs-clients** you can specify exactly which clients are allowed to connect to the NFS transport exports that are created on the secondary cluster. This parameter can be used for both inband and outband set up. Here is an example of the parameter being used for an inband setup:

- On the primary cluster, run the following command to configure independent fileset exports as AFM DR filesets and to backup configuration information:
  **mmcesdr primary config --output-file-path /root/ --ip-list "9.11.102.211,9.11.102.210" --rpo 10 --inband --allowed-nfs-clients --gateway-nodes**

This command will give the following output:

```
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.
```

## Creating the outband disaster recovery setup

Use the following steps for outband disaster recovery setup in an IBM Spectrum Scale cluster with protocols.

1. On the primary cluster, use the following command to configure independent fileset exports as AFM DR filesets and back up configuration information.

   ```
   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 the following:

   ```
   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
   ```

2. Use the following command to transfer the DR configuration file from the primary cluster to the secondary cluster.

   ```
   scp /root//DR_Config windwalker-vm1:/root/
   ```

   The system displays output similar to the following:

   ```
   root@windwalker-vm1's password:
   DR_Config 100% 2566 2.5KB/s 00:00
   ```

3. On the secondary cluster, use the following command to create the independent filesets that will later be paired with those on the primary cluster to form AFM DR pairs.

   ```
   mmcesdr secondary config --input-file-path /root --prep-outband-transfer
   ```

   The system displays output similar to the following:

```
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.

Transfer all data on primary cluster for fileset fs1:obj_sofpolicy2 to fileset fs1:obj_sofpolicy2 on
secondary cluster.

Transfer all data on primary cluster for fileset fs1:object_fileset to fileset fs1:object_fileset on
secondary cluster.

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.
```

4. Transfer the data from all protected filesets on the primary cluster to the corresponding filesets on the secondary cluster. If transferring files for which GPFS extended attributes also need to be transferred (such as object data), you must use a method that transfers GPFS extended attributes. For this purpose, you can use IBM Spectrum Protect (to back up data to tape and then restore it), GPFS cross-cluster mount, and AFM. Standard SCP and standard rsync do not transfer GPFS extended attributes.
5. After all the data has been transferred to the secondary cluster, use the following command to complete the setup on the secondary cluster.

   ```
   mmcesdr secondary config --input-file-path /root
   ```

   The system displays output similar to the following:

   ```
   Performing step 1/3, verification 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. Ensure that all of the expected AFM DR pairs show as `Active` in the output of the **mmafmctl** command and take corrective action if they do not.

   ```
   # mmafmctl fs0 getstate

   Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
   ------------ -------------- ------------- ------------ ------------ -------------
   nfs-ganesha1 nfs://9.11.102.210/gpfs/fs0/nfs-ganesha1 Active zippleback-vm2 0 4
   nfs-ganesha2 nfs://9.11.102.211/gpfs/fs0/nfs-ganesha2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 4
   combo1 nfs://9.11.102.210/gpfs/fs0/combo1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 7
   combo2 nfs://9.11.102.211/gpfs/fs0/combo2 Active zippleback-vm2 0 66
   smb1 nfs://9.11.102.210/gpfs/fs0/smb1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 65
   smb2 nfs://9.11.102.211/gpfs/fs0/smb2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 4

   # mmafmctl fs1 getstate

   Fileset Name Fileset Target Cache State Gateway Node Queue Length Queue numExec
   ------------ -------------- ------------- ------------ ------------ -------------
   object_fileset nfs://9.11.102.211/gpfs/fs1/object_fileset Active zippleback-vm1.tuc.stglabs.ibm.com 0 95671
   obj_sofpolicy1 nfs://9.11.102.211/gpfs/fs1/obj_sofpolicy1 Active zippleback-vm1.tuc.stglabs.ibm.com 0 27
   obj_sofpolicy2 nfs://9.11.102.210/gpfs/fs1/obj_sofpolicy2 Active zippleback-vm1.tuc.stglabs.ibm.com 0 26
   async_dr nfs://9.11.102.210/gpfs/fs1/.async_dr Active zippleback-vm1.tuc.stglabs.ibm.com 0 2751
   ```

**Note:** A state of `Dirty` is normal when data is actively being transferred from the primary cluster to the secondary cluster.

## Performing failover for protocols cluster when primary cluster fails

The failover procedure can use a new option to choose whether or not file protocols have their shares re-created or if the entire file protocol configuration for NFS and SMB is restored. The default is to re-create the NFS and SMB shares because it does not require removing and adding the file authentication again afterwards. The re-create option keeps existing SMB and NFS exports and customer data within their corresponding filesets, while the restore option deletes any existing SMB and NFS exports (but not data within their corresponding filesets) that were on secondary system before failover occurred. However, neither re-create nor restore affect object configuration or object data. The failover can be performed in one of the following ways:

## Re-create file export configuration

Use the following steps on the secondary cluster to fail over when the primary cluster fails in an IBM Spectrum Scale cluster with protocols.

On the secondary cluster, after the primary cluster has failed, use the following command.

```
mmcesdr secondary failover
```

The system displays output similar to the following:

```
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 restore.
Successfully completed step 3/4, protocol configuration restore.
Performing step 4/4, create/verify NFS AFM DR transport exports.
Successfully completed step 4/4, create/verify NFS AFM DR transport exports.
```

## Restore file export configuration

Use the following steps on the secondary cluster to fail over when the primary cluster fails in an IBM Spectrum Scale cluster with protocols.

1. On the secondary cluster, after the primary cluster has failed, use the following command.

   ```
   mmcesdr secondary failover --file-config --restore
   ```

   The system displays output similar to the following:

   ```
   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.

   ================================================================================
   = If all steps completed successfully, please remove and then re-create file
   = authentication on the DR cluster.
   = Once this is complete, Protocol Cluster Failover will be complete.
   ================================================================================
   ```

2. Remove the file authentication on the secondary cluster after failover and then add back the file authentication before failover is considered to be complete and client operations can resume, but point to the secondary cluster.

## Performing failback to old primary for protocols cluster

When failing back to the old primary, the file protocol configuration can either be re-created or restored on the old primary. The NFS transport exports on the secondary need to be re-created or NFS configuration can be restored. The re-create option keeps existing SMB and NFS exports and customer data within their corresponding filesets, while the restore option deletes any existing SMB and NFS exports (but not data within their corresponding filesets) that were on secondary system before failover occurred. However, neither re-create nor restore affect object configuration or object data.

## Re-create file protocol configuration for old primary

In the following example, file protocol configuration is re-created. To re-create the file exports on the old primary during restore, one of these commands can be run: **mmcesdr primary restore** or **mmcesdr primary restore --file-config --recreate**. The completion of failback on the secondary where the NFS transport export is re-created can also be performed by running one of these commands: **mmcesdr secondary failback --post-failback-complete** or **mmcesdr secondary failback --post-failback-complete --file-config --recreate**. Use the following steps for failing back to an old primary cluster in an IBM Spectrum Scale cluster with protocols.

1. On the old primary cluster, use the following command.

   ```
   mmcesdr primary failback --start --input-file-path "/root/"
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

   The system displays output similar to the following:

   ```
   Performing failback to primary on all AFM DR protected filesets.
   Successfully completed failback to primary on all AFM DR protected filesets
   ```

2. On the old primary cluster, use the following command one or more times until the amount of time it takes to complete the operation is less than the RPO value that you have set.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:object_fileset and is 0 Hrs. 25 Mins. 20 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before stopping
   failback.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

3. On the secondary cluster (acting primary), quiesce all client operations.
4. On the old primary cluster, use the following command one more time.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:object_fileset and is 0 Hrs. 0 Mins. 27 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before stopping
   failback.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

5. On the old primary cluster, use the following command.

   ```
   mmcesdr primary failback --stop --input-file-path "/root/"
   ```

   The system displays output similar to the following:

```
Performing stop of failback to primary on all AFM DR protected filesets.
Successfully completed stop failback to primary on all AFM DR protected filesets.
```

**Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

6. On the old primary cluster, use the following command to restore configuration:

   ```
   mmcesdr primary restore
   ```

   The system displays output similar to the following:

   ```
   Restoring cluster and enabled protocol configurations/exports.
   Successfully completed restoring cluster and enabled protocol configurations/exports.
   ```

7. On the secondary cluster (acting primary), use the following command to convert it back to a secondary cluster and associate it with the original primary cluster:

   ```
   mmcesdr secondary failback --post-failback-complete
   ```

   The system displays output similar to the following:

   ```
   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/recreating AFM DR-based NFS share configuration.
   Successfully completed step 2/2, restoring/recreating AFM DR-based NFS share configuration.

   ============================================================================
   = If all steps completed successfully, remove and then re-create file
   = authentication on the Secondary cluster.
   = Once this is complete, Protocol Cluster Failback will be complete.
   ============================================================================
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

## Restore file protocol configuration for old primary

In the following example, file protocol configuration is restored. To restore the file exports on the old primary during restore, run the following command: **mmcesdr primary restore --file-config --restore**. The completion of failback on the secondary where the NFS transport export is re-created can also be performed by running this commands: **mmcesdr secondary failback --post-failback-complete --file-config --restore**. Use the following steps for failing back to an old primary cluster in an IBM Spectrum Scale cluster with protocols.

1. On the old primary cluster, use the following command.

   ```
   mmcesdr primary failback --start --input-file-path "/root/"
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

   The system displays output similar to the following:

   ```
   Performing failback to primary on all AFM DR protected filesets.
   Successfully completed failback to primary on all AFM DR protected filesets
   ```

2. On the old primary cluster, use the following command one or more times until the amount of time it takes to complete the operation is less than the RPO value that you have set.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:object_fileset and is 0 Hrs. 25 Mins. 20 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before stopping
   failback.
   ```

**Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

3. On the secondary cluster (acting primary), quiesce all client operations.
4. On the old primary cluster, use the following command one more time.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:object_fileset and is 0 Hrs. 0 Mins. 27 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before stopping
   failback.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.
5. On the old primary cluster, use the following command.

   ```
   mmcesdr primary failback --stop --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing stop of failback to primary on all AFM DR protected filesets.
   Successfully completed stop failback to primary on all AFM DR protected filesets.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.
6. On the old primary cluster, use the following command to restore configuration:

   ```
   mmcesdr primary restore --file-config --restore
   ```

   The system displays output similar to the following:

   ```
   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.
   ================================================================================
   ```
7. On the primary cluster, remove the file authentication and then add it again.
8. On the secondary cluster (acting primary), use the following command to convert it back to a secondary cluster and associate it with the original primary cluster.

   ```
   mmcesdr secondary failback --post-failback-complete --input-file-path /root --file-config --restore
   ```

   The system displays output similar to the following:

   ```
   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.
   Successfully completed step 1/2, converting protected filesets back into AFM DR secondary filesets.
   Performing step 2/2, restoring/recreating AFM DR-based NFS share configuration.
   Successfully completed step 2/2, restoring/recreating AFM DR-based NFS share configuration.

   ================================================================================
   = If all steps completed successfully, remove and then re-create file
   = authentication on the Secondary cluster.
   = Once this is complete, Protocol Cluster Failback will be complete.
   ================================================================================
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.
9. On the secondary cluster, remove the file authentication and then add it again.

## Performing failback to new primary for protocols cluster

When failing back to the new primary, file protocol configuration can either be re-created or restored on the new primary. The NFS transport exports on the secondary need to be re-created or NFS configuration can be restored. The re-create option keeps existing SMB and NFS exports and customer data within their corresponding filesets, while the restore option deletes any existing SMB and NFS exports (but not data within their corresponding filesets) that were on secondary system before failover occurred. However, neither re-create nor restore affect object configuration or object data.

## Re-create file protocol configuration for new primary

The file protocol configuration is re-created in the following example. To re-create the file exports on the new primary during restore, one of these commands can be run:
**mmcesdr primary restore** or **mmcesdr primary restore --file-config --recreate**.
The completion of failback on the secondary where the NFS transport exports is re-created can also be performed by running one of these commands:
**mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path "/root"** or **mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path "/root" --file-config --recreate**.
Use the following steps for failing over to a new primary cluster in an IBM Spectrum Scale cluster with protocols.

1. On the old secondary cluster, use the following command to prepare recovery snapshots that contain data that will be transferred to the new primary cluster.

   ```
   mmcesdr secondary failback --generate-recovery-snapshots --output-file-path "/root/"
   --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   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
```

2. Transfer the newly created DR configuration file to the new primary cluster.

   ```
   scp /root//DR_Config zippleback-vm1:/root/
   ```

   The system displays output similar to the following:

   ```
   root@zippleback-vm1's password:
   DR_Config 100% 1996 2.0KB/s 00:00
   ```

3. On the new primary cluster, use the following command to create the independent filesets that will receive the data transferred from the recovery snapshots.

   ```
   mmcesdr primary failback --prep-outband-transfer --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Creating independent filesets to be used as recipients of AFM DR outband transfer of data.
   Successfully completed creating independent filesets to be used as recipients of AFM DR outband
   transfer of data.
   ```

Transfer data from recovery snapshots through outbound trucking to the newly created independent filesets before proceeding to the next step.

4. Transfer data from within the recovery snapshots of the secondary cluster to the new primary cluster.

   **Note:** Only one transfer is shown in the example below.

   ```
   rsync -av /gpfs/fs0/smb2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-6/*
   zippleback-vm1:/gpfs/fs0/smb2/
   ```

   The system displays output similar to the following:

   ```
   root@zippleback-vm1's password:
   sending incremental file list
   test

   sent 68 bytes received 31 bytes 15.23 bytes/sec
   total size is 0 speedup is 0.00
   ```

   **Attention:** When transferring files that need to also transfer GPFS extended attributes, extra steps are required. This example uses standard rsync which does not transfer extended attributes.

5. On the new primary cluster, use the following command to convert the independent filesets to primary filesets and generate a new DR configuration file that will be used on the primary cluster for the next steps and then transferred to the secondary cluster to be used in a later step.

   ```
   mmcesdr primary failback --convert-new --output-file-path /root/ --input-file-path /root/
   ```

   The system displays output similar to the following:

   ```
   Performing step 1/2, conversion of independent filesets into new primary filesets to be used for AFM DR.
   Successfully completed step 1/2, failback to primary on all AFM DR protected filesets.
   Performing step 2/2, creation of output file for remaining failback to new primary steps.
   Successfully completed step 2/2, creation of output file for remaining failback to new primary steps.

   File to be used with new primary cluster in next step of failback to new primary cluster: /root//DR_Config
   ```

6. On the new primary cluster, use the following command.

   ```
   mmcesdr primary failback --start --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing failback to primary on all AFM DR protected filesets.
   Successfully completed failback to primary on all AFM DR protected filesets.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

7. On the new primary cluster, use the following command one or more times until the amount of time it takes to complete the operation is less than the RPO value that you have set.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:obj_sofpolicy1 and is 0 Hrs. 45 Mins. 10 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before
   stopping failback.
   ```

8. On the secondary cluster (acting primary), quiesce all client operations.
9. On the new primary cluster, use the following command one more time.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:obj_sofpolicy1 and is 0 Hrs. 0 Mins. 16 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before
   stopping failback.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.
10. On the new primary cluster, use the following command to stop the failback process and convert the new primary filesets to read/write.

    ```
    mmcesdr primary failback --stop --input-file-path "/root/"
    ```

    The system displays output similar to the following:

    ```
    Performing stop of failback to primary on all AFM DR protected filesets.
    Successfully completed stop failback to primary on all AFM DR protected filesets.
    ```

11. On the new primary cluster, use the following command to restore the protocol and export services configuration information.

    ```
    mmcesdr primary restore --new-primary
    ```

    **Note:** The `--new-primary` option must be used to ensure protocol configuration is restored correctly.

    The system displays output similar to the following:

    ```
    Restoring cluster and enabled protocol
              configurations/exports.Successfully completed restoring cluster and enabled protocol
              configurations/exports.
    ```

12. Transfer the updated DR configuration file from the new primary cluster to the secondary cluster.

    ```
    scp /root//DR_Config windwalker-vm1:/root/
    ```

    The system displays output similar to the following:

    ```
    root@windwalker-vm1's password:
    DR_Config 100% 2566 2.5KB/s 00:00
    ```

13. On the secondary cluster, use the following command to register the new primary AFM IDs to the independent filesets on the secondary cluster acting as part of the AFM DR pairs.

    ```
    mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path "/root"
    ```

    The system displays output similar to the following:

    ```
    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, recreating AFM DR-based NFS share configuration.
    Successfully completed step 2/2, recreating AFM DR-based NFS share configuration.

    ================================================================================
    = If all steps completed successfully, remove and then re-create file
    ```

```
= authentication on the Secondary cluster.
= Once this is complete, Protocol Cluster Failback will be complete.
================================================================================
```

## Restore file protocol configuration for new primary

The file protocol configuration is restored in the following example. To restore the file exports on the old primary during restore, run the following command: **mmcesdr primary restore --file-config --restore**. The completion of failback on the secondary where the NFS transport export is re-created can also be performed by running this commands: **mmcesdr secondary failback --post-failback-complete --file-config --restore**. Use the following steps for failing back to an old primary cluster in an IBM Spectrum Scale cluster with protocols

1. On the old secondary cluster, use the following command to prepare recovery snapshots that contain data that will be transferred to the new primary cluster.

   ```
   mmcesdr secondary failback --generate-recovery-snapshots --output-file-path "/root/"
   --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   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
   ```

2. Transfer the newly created DR configuration file to the new primary cluster.

   ```
   scp /root//DR_Config zippleback-vm1:/root/
   ```

   The system displays output similar to the following:

   ```
   root@zippleback-vm1's password:
   DR_Config 100% 1996 2.0KB/s 00:00
   ```

3. On the new primary cluster, use the following command to create the independent filesets that will receive the data transferred from the recovery snapshots.

   ```
   mmcesdr primary failback --prep-outband-transfer --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Creating independent filesets to be used as recipients of AFM DR outband transfer of data.
   Successfully completed creating independent filesets to be used as recipients of AFM DR outband
   transfer of data.
   ```

Transfer data from recovery snapshots through outbound trucking to the newly created independent filesets before proceeding to the next step.

4. Transfer data from within the recovery snapshots of the secondary cluster to the new primary cluster.

   **Note:** Only one transfer is shown in the example below.

   ```
   rsync -av /gpfs/fs0/smb2/.snapshots/psnap0-newprimary-base-rpo-090B66F65623DEBF-6/*
   zippleback-vm1:/gpfs/fs0/smb2/
   ```

   The system displays output similar to the following:

   ```
   root@zippleback-vm1's password:
   sending incremental file list
   test

   sent 68 bytes received 31 bytes 15.23 bytes/sec
   total size is 0 speedup is 0.00
   ```

   **Attention:** When transferring files that need to also transfer GPFS extended attributes, extra steps are required. This example uses standard rsync which does not transfer extended attributes.

5. On the new primary cluster, use the following command to convert the independent filesets to primary filesets and generate a new DR configuration file that will be used on the primary cluster for the next steps and then transferred to the secondary cluster to be used in a later step.

   ```
   mmcesdr primary failback --convert-new --output-file-path /root/ --input-file-path /root/
   ```

   The system displays output similar to the following:

   ```
   Performing step 1/2, conversion of independent filesets into new primary filesets to be used for AFM DR.
   Successfully completed step 1/2, failback to primary on all AFM DR protected filesets.
   Performing step 2/2, creation of output file for remaining failback to new primary steps.
   Successfully completed step 2/2, creation of output file for remaining failback to new primary steps.

   File to be used with new primary cluster in next step of failback to new primary cluster: /root//DR_Config
   ```

6. On the new primary cluster, use the following command.

   ```
   mmcesdr primary failback --start --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing failback to primary on all AFM DR protected filesets.
   Successfully completed failback to primary on all AFM DR protected filesets.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

7. On the new primary cluster, use the following command one or more times until the amount of time it takes to complete the operation is less than the RPO value that you have set.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:obj_sofpolicy1 and is 0 Hrs. 45 Mins. 10 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before
   stopping failback.
   ```

8. On the secondary cluster (acting primary), quiesce all client operations.

9. On the new primary cluster, use the following command one more time.

   ```
   mmcesdr primary failback --apply-updates --input-file-path "/root/"
   ```

   The system displays output similar to the following:

   ```
   Performing apply updates on all AFM DR protected filesets.
   Longest elapsed time is for fileset fs1:obj_sofpolicy1 and is 0 Hrs. 0 Mins. 16 Secs.
   Successfully completed failback update on all AFM DR protected filesets.
   Depending on user load on the acting primary, this step may need to be performed again before
   stopping failback.
   ```

   **Note:** The **--input-file-path** parameter is optional but it might be needed if access to the configuration file is not available in the configuration fileset.

10. On the new primary cluster, use the following command to stop the failback process and convert the new primary filesets to read/write.

    ```
    mmcesdr primary failback --stop --input-file-path "/root/"
    ```

    The system displays output similar to the following:

    ```
    Performing stop of failback to primary on all AFM DR protected filesets.
    Successfully completed stop failback to primary on all AFM DR protected filesets.
    ```

11. On the new primary cluster, use the following command to restore the protocol and export services configuration information.

    ```
    mmcesdr primary restore --new-primary --file-config --restore
    ```

    **Note:** The `--new-primary` option must be used to ensure protocol configuration is restored correctly.

    The system displays output similar to the following:

    ```
    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.
    ================================================================================
    ```

12. On the primary cluster, remove the file authentication and then add it again.
13. Transfer the updated DR configuration file from the new primary cluster to the secondary cluster.

    ```
    scp /root//DR_Config windwalker-vm1:/root/
    ```

    The system displays output similar to the following:

    ```
    root@windwalker-vm1's password:
    DR_Config 100% 2566 2.5KB/s 00:00
    ```

14. On the secondary cluster, use the following command to register the new primary AFM IDs to the independent filesets on the secondary cluster acting as part of the AFM DR pairs.

    ```
    mmcesdr secondary failback --post-failback-complete --new-primary --input-file-path "/root"--file-config --restore
    ```

    The system displays output similar to the following:

    ```
    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.

    ================================================================================
    = If all steps completed successfully, remove and then re-create file
    = authentication on the Secondary cluster.
    = Once this is complete, Protocol Cluster Failback will be complete.
    ================================================================================
    ```

15. On the secondary cluster, remove the file authentication and then add it again.

## Backing up and restoring protocols and CES configuration information

The backup and restore capabilities of the **mmcesdr** command can be used when there is no secondary cluster and the user still wants to protect protocol and export services configuration information. The independent fileset used to store protocol and export services configuration information can be backed up using IBM Spectrum Protect software and then restored if it ever needs to be restored to the cluster.

**Note:** The following steps only describe how to back up and restore the protocols and CES configuration information. The actual data contained in protocol exports would need to be backed up and restored separately.

1. On the primary cluster, use the following command to back up the configuration information:

   ```
   mmcesdr primary backup
   ```

   The system displays output similar to the following:

   ```
   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.
   ```

   For backup, you can use IBM Spectrum Protect (formerly known as Tivoli Storage Manager) or some other tool. For example, you can use **mmbackup** as follows:

   ```
   mmbackup configuration_fileset_link_point --scope inodepace -t full
   ```

2. On the primary cluster, restore data from the off cluster storage into the configuration fileset. If **mmbackup** was used to back up the configuration fileset, the IBM Spectrum Protect command to restore is similar to the following:

   ```
   dsmc restore -subdir=yes "configuration_fileset_link_point /*"
   ```

3. On the primary cluster, use the following command to restore the configuration information:

   ```
   mmcesdr primary restore
   ```

   The system displays output similar to the following:

   ```
   Restoring cluster and enabled protocol configurations/exports.
   Successfully completed restoring cluster and enabled protocol configurations/exports.
   ```

   In some cases, running the **mmcesdr primary restore** command might display the following error message:
   `Saved configuration file does not exist.`. In this case, do the following:

   - If this cluster is part of a Protocols DR relationship, place a copy of the DR configuration file at a specified location and run the **mmcesdr primary restore** command again using the `--input-file-path` option.
   - If this cluster is not part of a Protocols DR relationship, run this command again with the `--file-config --restore` option to force restoring the file configuration information. The system displays output similar to the following:

     ```
     # mmcesdr primary restore --file-config --restore
     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.
     ================================================================================
     ```

4. On the primary cluster, remove the file authentication and then add it again:

   **Note:** If you want to perform a restore as part of a failback (either to an old primary cluster or a new primary cluster) and want to re-create the file configuration/exports, use one of the following commands:
   **mmcesdr primary restore**

or
**mmcesdr primary restore --file-config --recreate**.

## Updating protocols and CES configuration information

Protocol configuration information is backed up when primary configuration is run. However, if any of the protocol settings change after initial primary configuration, the saved configuration information must be updated.

You can use the following command to update the backed up configuration information for Object, NFS, and SMB protocols, and CES.

```
mmcesdr primary update {--obj | --nfs | --smb | --ces}
```

**Note:** No output is generated by the command, because this command is designed to be scripted and run on a regular basis or run as a part of a callback. The update command can only be used to update the primary configuration after Protocols DR has been configured. If no secondary cluster exists, use the **mmcesdr primary backup** command to back up configuration for later restore.

## Protocols and cluster configuration data required for disaster recovery

For protocols cluster disaster recovery, data needs to be collected for failover, failback, backup, or restore from the respective protocol, for authentication, and for cluster wide information.

Use the following information to collect the data required for protocols cluster disaster recovery.

## Object data required for protocols cluster DR

Data required for object protocol in case of a disaster recovery scenario is as follows.

**Note:** IBM Spectrum Scale 4.2 and later versions for object storage supports either AFM DR-based protection or multi-region object deployment , but not both. If multi-region object deployment is enabled, no object data or configuration information is protected through protocols cluster DR.

In addition to the standard object fileset, independent filesets are created for each object policy that is created. This in turn creates additional CCR files that are listed here. All of these additional filesets and additional configuration information are protected, if IBM Spectrum Scale for object storage is using the AFM DR-based protection and not multi-region object deployment.

You can determine all object filesets using the **mmobj policy list** command.

The object related files in CCR that need to be backed up are as follows:

1. `account.builder`
2. `account.ring.gz`
3. `account-server.conf`
4. `container.builder`
5. `container-reconciler.conf`
6. `container.ring.gz`
7. `container-server.conf`
8. `keystone_ssl.tar`
9. `keystone.tar`
10. `object.builder`
11. `object<index>.builder` for each storage policy, where <*index*> is a unique number representing the storage policy

12. `object-expirer.conf`
13. `object.ring.gz`
14. `object<index>.ring.gz` for each storage policy, where <*index*> is a unique number representing the storage policy
15. `object-server.conf`
16. `object-server-sof.conf`
17. `openrc`
18. `objRingVersion`
19. `postgresql-obj.service`
20. `proxy-server.conf`
21. `spectrum-scale-object.conf`
22. `spectrum-scale-object-policies.conf`
23. `spectrum-scale-objectizer.conf`
24. `spectrum-scale-local-region.conf`
25. `spectrum-scale-global-region.conf`
26. `spectrum-scale-compression-scheduler.conf`
27. `spectrum-scale-global-region.conf`
28. `spectrum-scale-compression-status.stat`
29. `swift.conf`
30. `swift.tar`

The following CCR files also need to be backed up FOR local object authentication :

- keystone.conf
- keystone-paste.ini
- logging.conf

For a list of object authentication related CCR files and variables that need to be backed up, see "Authentication related data required for protocols cluster DR" on page 437.

You can back up Postgres database files as follows:

1. Take the file system snapshot of the shared root file system.
2. Create a tar or a zip file of the `object` directory within the CES shared root file system and everything underneath this directory.
3. Save the tar or the zip file
4. Delete the snapshot.

## Failover steps for object configuration if you are using local authentication for object

Use the following steps on a protocol node in the secondary cluster to fail over the object configuration data. Use this set of steps if you are using local authentication for object.

You need to determine the following two values in the secondary cluster:

- Cluster host name: This is the DNS value which returns a CES IP address from the pool of addresses in the secondary cluster.
- Object database node: This is the CES IP address which is configured to run the postgresql-obj database for object services. You can find this value as the address designated as the `object_database_node` in the output of the **`mmces address list`** command. For example:

```
mmces address list

Address        Node      Group   Attribute
-------------------------------------------------------------
10.0.100.115   vwnode3   none    object_database_node
10.0.100.116   vwnode3   none    object_singleton_node
```

**Important:**

The following object steps must be run on the node designated as `object_database_node` in the secondary cluster. This ensures that postgresql-obj and Keystone servers can connect during this configuration process.

1. Stop the object protocol services using the following command:
   ```
   mmces service stop OBJ --all
   ```
2. Make two changes in the preserved Cluster Configuration Repository (CCR) configuration to update it for the DR environment:
   a. The `keystone.conf` file: Edit this file to change the database connection address to `object_database_node` of the secondary cluster. For example:
      ```
      Change this:
      [database]
      connection = postgresql://keystone:password@192.168.56.1/keystone

      to this:
      [database]
      connection = postgresql://keystone:password@192.168.1.3/keystone
      ```

      **Note:** If the **mmcesdr** command is used to save the protocol cluster configuration, then the preserved copy of the `keystone.conf` file is located at the following location:

      *CES_shared_root_mount_point*`/.async_dr/Failover_Config/Object_Config/latest/ccr_files/keystone.conf`

      You can edit the file directly to make this change or use the **openstack-config** command. For example, first retrieve the current value using `get`, and then update it using the `set` option:
      ```
      openstack-config --get keystone.conf  database connection \
      postgresql://keystone:passw0rd@192.168.56.1/keystone

      openstack-config --set keystone.conf  database connection \
      postgresql://keystone:passw0rd@192.168.1.3/keystone

      openstack-config --get keystone.conf  database connection \
      postgresql://keystone:passw0rd@192.168.1.3/keystone
      ```
   b. The *ks_dns_name* variable: This is one of the object related variables that was originally preserved from CCR. Modify the value of this variable to the cluster hostname of the secondary cluster.

      **Note:** If the **mmcesdr** command is used to save the protocol cluster configuration, then the preserved copy of the *ks_dns_name* variable is located as a line in the following file:

      *CES_shared_root_mount_point*`/.async_dr/Failover_Config/Object_Config/latest/ccr_vars/file_for_ccr_variables.txt`

      Change the value of the variable in this preserved copy of the file.
3. [Optional] If `spectrum-scale-localRegion.conf` exists from CCR, change the cluster hostname and cluster_id properties to the cluster host name and cluster id as shown in the output of the **mmlscluster** command.
4. Restore the Postgres database information to the shared root directory. The directory needs to be first cleaned out before the archive is restored. This can be done with commands similar to the following, assuming that the directory was tar/zip when it was backed up:

a. Delete the old Postgres data:
   ```
   rm -rf <shared_root_location>/object/keystone/*
   ```
b. Verify that the shared root directory is empty:
   ```
   ls <shared_root_location>
   ```
c. Restore the current Postgres database:
   ```
   tar xzf <tar_file_name>.gz -C <shared_root_location>
   ```
d. Delete the process status file from the primary:
   ```
   rm -rf <shared_root_location>/object/keystone/postmaster.pid
   ```
e. List the Postgres files:
   ```
   ls <shared_root_location>
   ```

5. Restore all object configuration CCR files, including **keystone.conf** with the modification for `object_database_node`, with a command similar to the following:
   ```
   mmccr fput <file> <location>/<file>
   ```
6. If object policies are present, restore all of the object policy related CCR files.
7. Restore all object configuration CCR variables, including *ks_dns_name* with the modification for cluster host name, with a command similar to the following:
   ```
   mmccr vput name value
   ```
8. Start the Postgres database and verify that it is running successfully using commands similar to the following:
   ```
   systemctl start postgresql-obj
   sleep 5
   systemctl status postgresql-obj
   ```

   **These commands generate output similar to:**
   ```
   postgresql-obj.service - postgresql-obj database server
   Loaded: loaded (/etc/systemd/system/postgresql-obj.service; disabled)
   Active: active (running) since Thu 2015-05-28 19:00:17 EDT; 20h ago
   ```
9. Load `openrc` definitions by running the following command.
   ```
   source /root/openrc
   ```
10. Run the following command for the value of the api_v3 pipeline. Save the output of this command into the variable <savedAPI_V3Pipeline>.
    `mmobj config list --ccrfile keystone-paste.ini --section pipeline:api_v3 --property pipeline` . If the value of the api_v3 pipeline does not contain the string **admin_token_auth**, then do the following:
    a. Make a note that the api_v3 pipeline had to be updated.
    b. Generate the new value of the api_v3 pipeline by inserting the string **admin_token_auth** directly after the string **token_auth** in the saved copy of the api_v3 pipeline.
    c. Change the api_v3 pipeline value using the command: `mmobj config change --ccrfile keystone-paste.ini --section pipeline:api_v3 --property pipeline --value <newAPI_V3Pipeline>`, where *<newAPI_V3Pipeline>* is the updated value from the previous step.
    d. List the updated api_v3 pipeline value by running the following command and ensure the **admin_token_auth** string is present: `mmobj config list --ccrfile keystone-paste.ini --section pipeline:api_v3 --property pipeline`.
11. Save the restored CCR copy of the `keystone.conf` to the local location using a command similar to **mmccr fget keystone.conf /etc/keystone/keystone.conf**. Also, update the owner and the group of this file using the following command: **chown keystone:keystone /etc/keystone/keystone.conf**.

    **Note:** If SSL is enabled, SSL certificates must be in place when you are saving `keystone.conf` from another cluster.
12. If the **DEFAULT admin_token** is set, save its current value by using a command similar to the following:

```
openstack-config --get /etc/keystone/keystone.conf DEFAULT admin_token
```

If a value is returned from the above command, save it because it will need to be restored later.

13. In the keystone.conf file, set **admin_token** to ADMIN using the **openstack-config** command as follows.

```
openstack-config --set /etc/keystone/keystone.conf DEFAULT admin_token ADMIN
```

14. Set the following environment variables.

```
export OS_TOKEN=ADMIN     # The value from admin_token
export OS_URL="http://127.0.0.1:35357/v3"
```

15. Start Keystone services and get the list of endpoint definitions using commands similar to the following.

```
systemctl start httpd
sleep 5
openstack endpoint list
```

**These commands generate output similar to:**

```
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
| ID         | Region | Service Name | Service Type | Enabled | Interface | URL                                                             |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
| c36e..9da5 | None   | keystone     | identity     | True    | public    | http://specscaleswift.example.com:5000/                         |
| f4d6..b040 | None   | keystone     | identity     | True    | internal  | http://specscaleswift.example.com:35357/                        |
| d390..0bf6 | None   | keystone     | identity     | True    | admin     | http://specscaleswift.example.com:35357/                        |
| 2e63..f023 | None   | swift        | object-store | True    | public    | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s    |
| cd37..9597 | None   | swift        | object-store | True    | internal  | http://specscaleswift.example.com:8080/v1/AUTH_%(tenant_id)s    |
| a349..58ef | None   | swift        | object-store | True    | admin     | http://specscaleswift.example.com:8080                          |
+------------+--------+--------------+--------------+---------+-----------+-----------------------------------------------------------------+
```

16. Update the host name specified in the endpoint definitions in the URL value from the endpoint list. The values in the endpoint table might have the cluster host name (ces1 in this example) from the primary system. They need to be updated for the cluster host name in the DR environment. In some environments, the cluster host name is the same between the primary and secondary clusters. If that is the case, skip this step.
    a. Delete the existing endpoints with the incorrect cluster host name. For each of the endpoints, use the ID value to delete the endpoint. For example, use a command similar to this to delete each of the six endpoints:

    ```
    openstack endpoint delete e149
    ```

    b. Recreate the endpoints with the cluster host name of the secondary cluster using the following commands:

    The *CHN* variable in the following commands is the cluster host name for the secondary cluster.

    ```
    openstack endpoint create identity public "http://$CHN:5000/v3"
    openstack endpoint create identity internal "http://$CHN:35357/v3"
    openstack endpoint create identity admin "http://$CHN:35357/v3"
    openstack endpoint create object-store public "http://$CHN:8080/v1/AUTH_%(tenant_id)s"
    openstack endpoint create object-store internal "http://$CHN:8080/v1/AUTH_%(tenant_id)s"
    openstack endpoint create object-store admin "http://$CHN:8080"
    ```

    c. Verify that the endpoints are now using the correct cluster host name using the following command:

    ```
    openstack endpoint list
    ```

17. If the api_v3 pipeline had to be updated previously, return it to its original value by running the following command: `mmobj config change --ccrfile keystone-paste.ini --section pipeline:api_v3 --property pipeline --value <savedAPI_V3Pipeline>`
, where *<savedAPI_V3Pipeline>* is the value of the api_v3 pipeline saved above.
18. Depending on whether a value for the DEFAULT admin_token was previously set, do one of the following:
    - If a value for the **DEFAULT admin_token** was previously set, reset it to that value using a command similar to the following:

    ```
    openstack-config --set /etc/keystone/keystone.conf DEFAULT admin_token ${currentAdminToken}
    ```

    - If there was no previous value for the **DEFAULT admin_token**, delete the value that was set to convert the endpoint definitions using a command similar to the following:

    ```
    openstack-config --del /etc/keystone/keystone.conf DEFAULT admin_token
    ```

19. Stop the services that were manually started earlier and clean up using the following commands:
```
systemctl stop httpd
systemctl stop postgresql-obj
unset OS_URL
unset OS_TOKEN
```
20. Start the object protocol services using the following command:
```
mmces service start OBJ --all
```

**Failover steps for object configuration if you are not using local authentication for object:**

Use the following steps on a protocol node on a secondary cluster to fail over object configuration data. Use this set of steps if you are not using local authentication for object.

1. Stop the object protocol services using the following command:
```
mmces service stop OBJ --all
```
2. Restore the Postgres database information to the shared root directory. The directory needs to be first cleaned out before the archive is restored. This can be done with commands similar to the following, assuming that the directory was tar/zip when it was backed up:
   a. Delete the old Postgres data:
   ```
   rm -rf <shared_root_location>/object/keystone/*
   ```
   b. Verify that the shared root directory is empty:
   ```
   ls <shared_root_location>
   ```
   c. Restore the current Postgres database:
   ```
   tar xzf <tar_file_name>.gz -C <shared_root_location>
   ```
   d. Delete the process status file from the primary:
   ```
   rm -rf <shared_root_location>/object/keystone/postmaster.pid
   ```
   e. List the Postgres files:
   ```
   ls <shared_root_location>
   ```
3. Restore all object configuration CCR files with a command similar to the following:
```
mmccr fput <file> <location>/<file>
```
4. If object policies are present, restore all of the object policy related CCR files.
5. Restore all object configuration CCR variables with a command similar to the following:
```
mmccr vput name value
```
6. Start the object protocol services using the following command:
```
mmces service start OBJ --all
```

## Failback or restore steps for object configuration

Use the following steps on a protocol node in the primary cluster to fail back or restore the object configuration data.

You need to determine the `object_database_node` on the primary cluster once repaired or replaced.

- Object database node: This is the CES IP address which is configured to run the postgresql-obj database for object services. You can find this value as the address designated as the `object_database_node` in the output of the **`mmces address list`** command. For example:
```
mmces address list

Address        Node      Group   Attribute
-------------------------------------------------------------
10.0.100.115   vwnode3   none    object_database_node
10.0.100.116   vwnode3   none    object_singleton_node
```

**Important:**

The following object steps must be run on the node designated as `object_database_node` in the primary cluster. This ensures that postgresql-obj and Keystone servers can connect during this configuration process.

1. Stop the object protocol services using the following command:
   ```
   mmces service stop OBJ --all
   ```
2. Restore the Postgres database information to the shared root directory. The directory needs to be first cleaned out before the archive is restored. This can be done with commands similar to the following, assuming that the directory was tar/zip when it was backed up:
   a. Delete the old Postgres data:
      ```
      rm -rf <shared_root_location>/object/keystone/*
      ```
   b. Verify that the shared root directory is empty:
      ```
      ls <shared_root_location>
      ```
   c. Restore the current Postgres database:
      ```
      tar xzf <tar_file_name>.gz -C <shared_root_location>
      ```
   d. Delete the process status file from the primary:
      ```
      rm -rf <shared_root_location>/object/keystone/postmaster.pid
      ```
   e. List the Postgres files:
      ```
      ls <shared_root_location>
      ```
3. Restore all object configuration CCR files with a command similar to the following:
   ```
   mmccr fput <file> <location>/<file>
   ```
4. If object policies are present, restore all of the object policy related CCR files.
5. Restore all object configuration CCR variables with a command similar to the following:
   ```
   mmccr vput name value
   ```
6. Start the object protocol services using the following command:
   ```
   mmces service start OBJ --all
   ```

## SMB data required for protocols cluster DR

Data required for the SMB protocol in case of a disaster recovery scenario is as follows.

You can determine the SMB shares using the **mmsmb export list** command.

The SMB protocol related files that need to be backed up are as follows.

- `account_policy.tdb`
- `autorid.tdb`[1]
- `group_mapping.tdb`
- `passdb.tdb`
- `registry.tdb`
- `secrets.tdb`
- `share_info.tdb`
- `ctdb.tdb`

**Note:** [1] This file is required only if the file authentication is configured with Active Directory.

The following information is common for all of these files:

- The type of these files is persistent TDB.
- They are not in the cluster configuration repository (CCR).
- Their location is `/var/lib/ctdb/persistent` on all protocol nodes.
- Their contents are same on all the nodes.

- Their name consists of TDB name + node specific extension. For example: `registry.tdb.0`

The private Kerberos configuration files available at the following location also need to be backed up: `/var/lib/samba/smb_krb5/`. You can copy these files from this location and save them.

## Failover steps for the SMB protocol

Use the following steps on a protocol node in the secondary cluster to fail over the SMB protocol configuration.

1. Before stopping the SMB services, make a note of the node number that will be used to restore TDB files because the node numbers are not available when SMB services are stopped.
2. Stop the SMB services using the following command:
   ```
   mmces service stop SMB --all
   ```
3. Issue the following command to stop the NFS service:
   ```
   mmces service stop NFS --all
   ```
4. Remove the contents of the `/var/lib/ctdb/persistent` directory on all protocol nodes.
5. Restore the previously saved TDB files to one of the protocol nodes and place them in the `/var/lib/ctdb/persistent` directory for the node where the node number was saved.

   However, when copying the files to that directory on the node, replace the `X.bak`, where X represents the node number where the files were copied from, with the new node number. It is crucial that each of these files ends with `.tdb.Y`, where Y is the node number that was saved and the node number where the files are being restored. These files only need to be put into one of the nodes and when the SMB processes are started again they are copied around to the other nodes properly.
6. Remove the contents of the `/var/lib/samba/smb_krb5` directory on all the protocol nodes.
7. Restore the saved contents of `smb_krb5` to the `/var/lib/samba/smb_krb5/` directory on one of the protocol nodes. No special extension needs to be altered in this case.
8. Start the SMB services using the following command:
   ```
   mmces service start SMB --all
   ```
9. Issue the following command to start the NFS service:
   ```
   mmces service start NFS --all
   ```
10. Remove SMB exports that are not protected using AFM DR independent filesets.

## Failback or restore steps for the SMB protocol

Use the following steps on a protocol node in the primary cluster, once repaired or replaced, to fail back or restore the SMB protocol configuration.

1. Stop the NFS services using the following command:
   ```
   mmces service stop NFS --all
   ```
2. Stop the SMB services using the following command:
   ```
   mmces service stop smb --all
   ```
3. Delete all files from the `/var/lib/ctdb/persistent` directory on all protocol nodes.
4. Restore all required persistent TDB files from the saved configuration location to the `/var/lib/ctdb/persistent` directory on one of the protocol nodes. Ensure that you append the node number to the end of file names.
5. Delete all private Kerberos configuration files in the `/var/lib/samba/smb_krb5/` directory on all protocol nodes.
6. Restore private Kerberos configuration files to the `/var/lib/samba/smb_krb5/` directory on one of the protocol nodes.
7. On the failback cluster, start the SMB services using the following command:
   ```
   mmces service start smb --all
   ```
8. Issue the following command on the failback cluster to start the NFS service:
   ```
   mmces service start NFS --all
   ```

## NFS data required for protocols cluster DR

Data required for NFS protocol in case of a disaster recovery scenario is as follows.

To find the NFS exports, enter the following command:

```
mmnfs export list
```

The system displays output similar to the following:

```
Path           Delegations Clients
---------------------------------------
/ibm/fs1/fset1 none        10.0.0.1
/ibm/fs1/fset1 none        10.0.0.2
/ibm/fs1/fset1 none        *
```

If the NFS exports are independent filesets, AFM based Disaster Recovery (AFM DR) can be used to replicate the data.

The NFS protocol related CCR files that need to be backed up are as follows.

- `gpfs.ganesha.main.conf`
- `gpfs.ganesha.nfsd.conf`
- `gpfs.ganesha.log.conf`
- `gpfs.ganesha.exports.conf`
- `gpfs.ganesha.statdargs.conf`

The following NFS protocol related CCR variable needs to be backed up.

- *nextexportid*

### Failover steps for the NFS protocol

Use the following steps on one of the protocol nodes in the secondary cluster to fail over the NFS protocol configuration.

1. Stop the NFS services using the following command:
   ```
   mmces service stop NFS --all
   ```
2. Edit the saved `gpfs.ganesha.exports.conf` export configuration file to remove all exports that are not protected through AFM DR independent filesets.
3. Restore the NFS related CCR files. For a list of these files, see "NFS data required for protocols cluster DR"
4. Restore the *nextexportid* CCR variable.
5. Load the exports file using the following command:
   ```
   mmnfs export load /<Path_to_CCR_files>/gpfs.ganesha.exports.conf
   ```
6. Start the NFS services using the following command:
   ```
   mmces service start NFS --all
   ```

### Failback or restore steps for the NFS protocol

Use the following steps on a protocol node in the primary cluster, once repaired or replaced, to fail back or restore the NFS protocol configuration.

1. Stop the NFS services using the following command:
   ```
   mmces service stop NFS --all
   ```
2. Restore the NFS related CCR files. For a list of these files, see "NFS data required for protocols cluster DR."
3. Restore the *nextexportid* CCR variable.
4. Load the exports file using the following command:
   ```
   mmnfs export load /<Path_to_saved_CCR_files>/gpfs.ganesha.exports.conf
   ```

5. Start the NFS services using the following command:
   ```
   mmces service start NFS --all
   ```

## Authentication related data required for protocols cluster DR

Authentication data required in case of a disaster recovery scenario is as follows.

The following authentication related CCR file needs to be backed up for disaster recovery.

- `authccr`

### File authentication related data

The following CCR variable needs to be backed up for file authentication:

- *FILE_AUTH_TYPE*

The following CCR variable is used by callbacks on each protocol node for authentication configuration:

- *KS_EXT_CACERT*

The following file is present when external keystone is configured with SSL:

- `/etc/swift/ks_ext_cacert.pem`

Depending on the file authentication scheme you are using, additional files need to be backed up.

**LDAP for file authentication:**

- `SSSD_CONF`
- `LDAP_CONF`
- `KRB5_CONF` [1]
- `KRB5_KEYTAB` [1]
- `LDAP_TLS_CACERT` [1]

**Active Directory (AD) for file authentication:**

- `KRB5_CONF`
- `KRB5_KEYTAB` [1]

**NIS for file authentication:**

- `SSSD_CONF`
- `YP_CONF`

**Note:** [1] This file is not always present.

### Object authentication related data

The object authentication related files in CCR that need to be backed up are as follows:

- `keystone.conf`
- `keystone-paste.ini`
- `logging.conf`
- `wsgi-keystone.conf`

The object authentication related variables in CCR that need to be backed up are as follows:

- *OBJECT_AUTH_TYPE*
- *PREV_OBJECT_AUTH_TYPE*

This variable may not be present if the authentication type has not changed.

- *OBJECT_IDMAPDELETE*
- *ks_db_type*
- *ks_db_user*
- *ks_db_user_pwd*
- *ks_dns_name*

### Failover steps for authentication data

Use the following steps on a protocol node in the secondary cluster to fail over the authentication configuration.

**Note:** The object authentication does not need to be removed and re-added as part of failover, failback, or restore.

1. Save the current file authentication information on the secondary cluster.
2. Remove file authentication from the secondary cluster.
3. Restore file authentication on the secondary cluster based on the information saved in step 1.

### Failback steps for authentication data

Use the following steps on a protocol node in the primary cluster, once repaired or replaced, to fail back the authentication configuration.

**Note:** The object authentication does not need to be removed and re-added as part of failover, failback, or restore.

1. Save the current file authentication information on the primary cluster.
2. Remove file authentication from the primary cluster.
3. Restore file authentication on the primary cluster based on the information saved in step 1.

**Restore steps for authentication data:**

Use the following steps on a protocol node in the primary cluster and the secondary cluster to restore the authentication configuration.

**Note:** The object authentication does not need to be removed and re-added as part of failover, failback, or restore.

1. Save the current file authentication information on the primary cluster.
2. Remove file authentication from the primary cluster.
3. Restore file authentication on the primary cluster based on the information saved in step 1.
4. Save the current file authentication information on the secondary cluster.
5. Remove file authentication from the secondary cluster.
6. Restore file authentication on the secondary cluster based on the information saved in step 4.

## CES data required for protocols cluster DR

Cluster Export Services (CES) data required in case of a disaster recovery scenario is as follows.

Cluster Configuration Repository (CCR) files that need to be backed up for CES in a disaster recovery scenario are as follows:

- `mmsdrfs`
- `cesiplist`

## Failover steps for CES

No Cluster Export Services (CES) configuration information is restored on fail over. This is because this information is typically cluster specific and it would interfere with the proper operating of the secondary cluster.

## Failback or recovery steps for CES

Use the following steps on a protocol node in the primary cluster to fail back or recover the CES configuration.

1. Restore the `cesiplist` file.
2. For each protocol node listed in the stored, backup copy of the `mmsdrfs` file, verify that the node on the primary cluster is also configured as a protocol node. If not, use the **mmchnode** to enable the node as a protocol node.
3. For each of the following CES parameters in the stored, backup copy of the `mmsdrfs` file, verify that the value is the same on the primary cluster. If not, use the **mmchconfig** to update the configuration value.
   - **cesSharedRoot**
   - **cesAddressPool**
   - **cesServices**
   - **cifsBypassTraversalChecking**
   - **syncSambaMetadataOps**
   - **cifsBypassShareLocksOnRename**

# Chapter 31. File Placement Optimizer

GPFS File Placement Optimizer (FPO) is a set of features that allow GPFS to operate efficiently in a system based on a shared nothing architecture. It is useful for big data applications that process massive amounts of data.

**Note:** This feature is available with IBM Spectrum Scale Express Edition, Standard Edition or higher.

FPO uses the following entities and policies:

**Chunks**

A chunk is a logical grouping of blocks that allows the grouping to behave like one large block, useful for applications that need high sequential bandwidth. Chunks are specified by a block group factor that dictates how many file system blocks are laid out sequentially on disk to behave like a large block. Different Chunk size can be defined by block group factor on file level or defined globally on a storage pool by default.

On the file level, the block group factor can be specified by the **--block-group-factor** argument of the **mmchattr** command. You can also specify the block group factor by the **setBGF** argument of the **mmchpolicy** and **mmapplypolicy** command.The range of the block group factor is 1 - 1024. The default value is 1. You can also specify the block group factor through the **blockGroupFactor** argument in a storage pool stanza (as input to the **mmadddisk** or **mmcrfs** command).

The effective chunk size is a multiplication of Block Group Factor and GPFS block size. For example, setting block size to 1 MB and block group factor to 128 leads to an effective large block size of 128 MB.

See the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmadddisk**
- **mmchattr**
- **mmcrfs**
- **mmchpolicy**
- **mmapplypolicy**

**Extended failure groups**

A failure group is defined as a set of disks that share a common point of failure that might cause them all to become simultaneously unavailable. Traditionally, GPFS failure groups are identified by simple integers. In an FPO-enabled environment, a failure group might be specified as not just a single number, but as a vector of up to three comma-separated numbers. This vector conveys topology information that GPFS exploits when making data placement decisions.

In general, a topology vector is a way for the user to specify which disks are closer together and which are farther away. In practice, the three elements of the failure group topology vector might represent the rack number of a disk, a position within the rack, and a node number. For example, the topology vector 2,1,0 identifies rack 2, bottom half, first node.

Also, the first two elements of the failure group represent the failure group ID and the three elements together represent the locality group ID. For example, 2,1 is the failure group ID and 2,1,0 is the locality group ID for the topology vector 2,1,0.

The Data block placement decisions about the disk selection for data replica are made by GPFS based on the Failure group. When considering two disks for striping or replica placement purposes, it is important to understand the following:

- Disks that differ in the first of the three numbers are farthest apart (as they are in different racks).
- Disks that have the same first number but differ in the second number are closer (as they are in the same rack, but in different halves).
- Disks that differ only in the third number reside in different nodes in the same half of the same rack.
- Only disks that have all three numbers in common reside in the same node.

The data block placement decisions are also affected by the level of replication and the value of the **writeAffinityDepth** parameter. For example, when using replication 3, GPFS might place two replicas far apart (different racks) to minimize chances of losing both. However, the third replica can be placed close to one of the others (same rack, but different half), to reduce network traffic between racks when writing the three replicas.

To specify the topology vector that identifies a failure group, you use the **failureGroup=***FailureGroup* attribute in an NSD stanza (as input to the **mmadddisk** or **mmcrfs** command).

See the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmadddisk**
- **mmcrfs**

**Write affinity depth**

Write affinity depth is a policy that allows the application to determine the layout of a file in the cluster to optimize for typical access patterns. The write affinity is specified by a depth that indicates the number of localized copies (as opposed to wide striped). It can be specified at the storage pool or file level. The enabling of Write affinity depth, indicates that the first replica is being written on the node where the writing is triggered. It also indicates, the second and third replica (if any) are being written on the other node disks.

To specify write affinity depth, you use the **writeAffinityDepth** attribute in a storage pool stanza (as input to the **mmadddisk** or **mmcrfs** command) or the **--write-affinity-depth** argument of the **mmchattr** command. You can use **--block-group-factor** argument of the **mmchpool** command to change a storage pool's block group factor. You can change write affinity depth by **--write-affinity-depth** argument of **mmchpool** for a storage pool. You can also specify the write affinity depth for file by the **setWAD** argument of the **mmchpolicy** and **mmapplypolicy** commands.

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.

**Note:** In fileset level, Write affinity depth of 2 is design to assign (write) all the files in a fileset to the same second-replica node. However, this behavior depends on node status in the cluster. After a node is added to or deleted from a cluster, a different node might be selected as the second replica for files in a fileset.

See the description of storage pool stanzas that follows. Also, see the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmadddisk**
- **mmchattr**
- **mmcrfs**
- **mmchpolicy**
- **mmapplypolicy**
- **mmchpool**

**Write affinity failure group**

Write affinity failure group is a policy that indicates the range of nodes (in a shared nothing architecture) where replicas of blocks in a particular file are to be written. The policy allows the application to determine the layout of a file in the cluster to optimize for typical access patterns.

You specify the write affinity failure group through the **write-affinity-failure-group** *WafgValueString* attribute of the **mmchattr** command. You can also specify write affinity failure group through the **setWADFG** attribute of the **mmchpolicy** and **mmapplypolicy** command. Failure group topology vector ranges specify the nodes, and the specification is repeated for each replica of the blocks in a file.

For example, the attribute `1,1,1:2;2,1,1:2;2,0,3:4` indicates:

- The first replica is on rack 1, rack location 1, nodes 1 or 2.
- The second replica is on rack 2, rack location 1, nodes 1 or 2.
- The third replica is on rack 2, rack location 0, nodes 3 or 4.

The default policy is a null specification. This default policy indicates that each replica must follow the storage pool or the file-write affinity depth (WAD) definition for data placement. Not wide striped over all disks.

When data in an FPO pool is backed up in a IBM Spectrum Protect server and then restored, the original placement map is broken unless you set the write affinity failure group for each file before backup.

**Note:** To change the failure group of a disk in a write-affinity–enabled storage pool, you must use the **mmdeldisk** and **mmadddisk** commands. You cannot use **mmchdisk** to change it directly.

See the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmchpolicy**
- **mmapplypolicy**
- **mmchattr**

**Enabling the FPO features**

To efficiently support write affinity and the rest of the FPO features, GPFS internally requires the creation of special allocation map formats. When you create a storage pool that is to contain files that make use of FPO features, you must specify **allowWriteAffinity=yes** in the storage pool stanza.

To enable the policy to read from preferred replicas, issue one of the following commands:

- To specify that the policy read from the first replica, regardless of whether there is a replica on the disk, default to or issue the following:

  ```
  mmchconfig readReplicaPolicy=default
  ```

- To specify that the policy read replicas from the local disk, if the local disk has data, issue the following:

  ```
  mmchconfig readReplicaPolicy=local
  ```

- To specify that the policy read replicas from the fastest disk to read from based on the disk's read I/O statistics, run the following:

  ```
  mmchconfig readReplicaPolicy=fastest
  ```

**Note:** In an FPO-enabled file system, if you run data locality awareness workload over FPO, such as Hadoop or Spark, configure **readReplicaPolicy** as *local* to read data from the local disks to reduce the network bandwidth consumption.

See the description of storage pool stanzas that follows. Also, see the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmadddisk**
- **mmchconfig**
- **mmcrfs**

**Storage pool stanzas**

*Storage pool stanzas* are used to specify the type of layout map and write affinity depth, and to enable write affinity, for each storage pool.

Storage pool stanzas have the following format:

```
%pool:
  pool=StoragePoolName
  blockSize=BlockSize
  usage={dataOnly | metadataOnly | dataAndMetadata}
  layoutMap={scatter | cluster}
  allowWriteAffinity={yes | no}
  writeAffinityDepth={0 | 1 | 2}
  blockGroupFactor=BlockGroupFactor
```

See the following command descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmadddisk**
- **mmcrfs**
- **mmchpool**

**Recovery from disk failure**

A typical shared nothing cluster is built with nodes that have direct-attached disks. Disks are not shared between nodes as in a regular GPFS cluster, so if the node is inaccessible, its disks are also inaccessible. GPFS provides means for automatic recovery from these and similar common disk failure situations.

The following command sets up and activates the disk recovery features:

```
mmchconfig restripeOnDiskFailure=yes -i
```

Whether a file system went through a recovery 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 run. The recovery actions are asynchronous and GPFS continues its processing while the recovery attempts take place. The results from the recovery actions and any errors that are encountered are recorded in the GPFS logs.

Two more parameters are available for fine-tuning the recovery process:

```
mmchconfig metadataDiskWaitTimeForRecovery=seconds
mmchconfig dataDiskWaitTimeForRecovery=seconds
```

The default value for **metadataDiskWaitTimeForRecovery** is 1800 seconds. The default value for **dataDiskWaitTimeForRecovery** is 3600 seconds.

See the following command description in the *IBM Spectrum Scale: Command and Programming Reference*:

- **mmchconfig**

## Distributing data across a cluster

You can distribute data uniformly across a cluster.

Following are the possible ways to distribute the data:

- Import the data through a node that does not have any attached NSD and takes the role as a GPFS client node in the cluster. This ensures that the data is distributed evenly across all failure groups and all nodes within a failure group.
- Use a write affinity depth of 0 across the cluster.
- Make every GPFS node an ingest node and deliver data equally across all ingest nodes. However, this strategy is expensive in terms of implementation.

Ideally, all the failure groups must have an equal number of disks with roughly equal capacity. If one failure group is much smaller than the rest, it is likely to fill up faster than the others, and this complicates rebalancing actions.

After the initial ingesting of data, the cluster might be unbalanced. In such a situation, use the **mmrestripefs** command with the **-b** option to rebalance the data.

**Note:** For FPO users, the **mmrestripefs -b** command breaks the original data placement that follows the data locality rule.

## FPO pool file placement and AFM

For AFM home or cache, an FPO pool file that is written on the local side is placed according to the write affinity depth and write affinity failure group definitions of the local side.

When a file is synced from home to cache, it follows the same FPO placement rule as when written from the gateway node in the cache cluster. When a file is synced from cache to home, it follows the same FPO data placement rule as when written from the NFS server in the home cluster.

To retain the same file placement at AFM home and cache, ensure that each has the same cluster configuration and set the write affinity failure group for each file. If the home and cache cluster have different configurations, such as the disk number, node number, or fail group, then the data locality might be broken.

## Configuring FPO

Follow the steps listed in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide* to install the GPFS RPMs and build the portability layer on all nodes in the cluster. You can configure password-less SSH for root user across all GPFS nodes. However, in cases of special security control, you can configure at least one node for the root user to access all GPFS nodes in a password-less mode. GPFS commands can be run only over these nodes.

For OS with Linux kernel 2.6, enter following commands on all GPFS nodes that are set as root to set vm.min_free_bytes :

```
# TOTAL_MEM=$(cat /proc/meminfo | grep MemTotal | tr -d \"[:alpha:]\" | tr -d \"[:punct:]\" | tr
-d \"[:blank:]\") # VM_MIN_FREE_KB=$((${TOTAL_MEM}*6/100))
```

```
# echo "vm.min_free_kbytes = $VM_MIN_FREE_KB" >> /etc/sysctl.conf # sysctl -p
# sysctl -a | grep vm.min_free_kbytes
```

# Configuring IBM Spectrum Scale Clusters

All GPFS configuration steps must be performed as the root user. These steps need to be executed only on one node, not on all nodes.

## Create the GPFS Cluster

The GPFS node file defines all nodes in the cluster and some of the roles.

Create the GPFS cluster with **node11** as the primary and **node21** as the secondary cluster configuration server. Set the *-A* flag to automatically start GPFS daemons when the OS is started.

```
# mmcrcluster -A -C gpfs-cluster -p node11 -s node21 -N nodefile -r $(which ssh) -R $(which scp)
```

Use the **mmlscluster** command to view the cluster.

## Apply IBM Spectrum Scale license

All GPFS nodes require a license designation before they can be used. The FPO feature introduced a dedicated PFS license class **fpo**. In a GPFS FPO cluster, all quorum and manager nodes require a server license. Based on the sample environment, **node11**, **node21**, and **node31** require a server license. The other nodes require an **fpo** license.

```
# mmchlicense server --accept -N node11,node21,node31
# mmchlicense fpo --accept -N node12,node13,node14,node15,node16,node22,node23,node24,node25,node26,
node32,node33,node34,node35,node36
```

Use the **mmlslicense -L** command to view license information for the cluster.

Nodes with no disks in the file system are called as diskless nodes. Run the **mmchlicense client --accept -N** command to accept the client license for disks that have no disks in the GPFS file system.

Start the GPFS cluster to verify whether it starts successfully. Use the **mmstartup** *–a* command to start the GPFS cluster and the **mmgetstate** *–a* command to view the state of the GPFS cluster.

## Create GPFS Network Shared Disks (NSD)

To create the network shared disks (NSD) in GPFS, create a disk file to be used as input to the **mmcrnsd** command. The disk file defines the GPFS pools and the NSDs. A recommended GPFS pool configuration has two storage pools, a system pool for metadata only and a data pool.

- Storage Pools
  - System pool contains all of the metadata disks and does not have FPO behavior enabled. The system pool should have a smaller block size than the data pool for performance reasons. If you choose to use **dataAndMetadata** disks in the system pool, you must set the system pool block size to be the same as the data pool block size as both the pools can have data. For the **dataAndMetadata** system pool, the block size 1M is recommended.
  - Data pool contains all of the data disks and has FPO behavior enabled by setting **allowWriteAffinity**=yes, **writeAffinityDepth**=1, and **blockGroupFactor**=128.

    The chunk size can be calculated as **blockSize** * **blockGroupFactor**. Similar to the HDFS recommendation, the GPFS FPO recommendation is **blockSize=2M * blockGroupFactor=64** for a chunk size of 128 MB
- NSD
  - Every local disk to be used by GPFS must have a matching entry in the disk stanza file
  - The device must match the device path of the disk.

    **Note:** In the example, `/dev/sda` is not included because this is the OS disk.

If **MapReduce** intermediate and temporary data is stored on etx3/ext4 disks instead of GPFS, make sure those disks are not included in the disk file or GPFS will format them and include them in the GPFS cluster

– System pool disks:
  - Should have **usage=metadataOnly**. It is possible to use **usage=dataAndMetadata** if there is a reason to have data on the system pool disks. The block size of the dataAndMetadata system pool must be the same as the block size of a data pool in the file system.
  - **failureGroup** must be a single number if **allowWriteAffinity** is not enabled (specify **allowWriteAffinity=no** for system pool definition when doing **mmcrnsd** or **mmcrfs**) and it should be the same for all disks on the same node. If **allowWriteAffinity** is enabled for system pool, the failure group can be of format *rack,position,node*, for example, **2,0,1**; or, it can take the traditional single-number failure group format also.
  - Even when **allowWriteAffinity** is enabled for system pool, the metadata does not follow data locality rules; these rules apply only to data placement
– Data pool disks:
  - Must have **usage**=dataOnly.
  - **failureGroup** must be of the format [rack,position,node], where position is either 0 or 1 to represent top or bottom half of the rack. The sample environment does not have half racks, so the same position is used for all nodes. Especially, when position and node fields are ignored in the cluster, the failure group can be defined as a single number, [rack, -,-].

Example of NSD disk file created by using the **mmcrnsd** command:

```
%pool: pool=system  blockSize=256K layoutMap=cluster allowWriteAffinity=no
%pool: pool=datapool blockSize=2M layoutMap=cluster allowWriteAffinity=yes writeAffinityDepth=1
blockGroupFactor=256

# gpfstest9
%nsd: nsd=node9_meta_sdb  device=/dev/sdb servers=gpfstest9 usage=metadataOnly failureGroup=1 pool=system

%nsd: nsd=node9_data_sdf2  device=/dev/sdf servers=gpfstest9 usage=dataOnly failureGroup=1,0,1 pool=datapool
%nsd: nsd=node9_data_sdg2  device=/dev/sdg servers=gpfstest9 usage=dataOnly failureGroup=1,0,1 pool=datapool

#gpfstest10
%nsd: nsd=node10_meta_sda  device=/dev/sda servers=gpfstest10 usage=metadataOnly failureGroup=2 pool=system

%nsd: nsd=node10_data_sde2  device=/dev/sde servers=gpfstest10 usage=dataOnly failureGroup=2,0,1 pool=datapool
%nsd: nsd=node10_data_sdg2  device=/dev/sdg servers=gpfstest10 usage=dataOnly failureGroup=2,0,1 pool=datapool

#gpfstest11

%nsd: nsd=node11_meta_sdb  device=/dev/sdb servers=gpfstest11 usage=metadataOnly failureGroup=3 pool=system

%nsd: nsd=node11_data_sdf2  device=/dev/sdf servers=gpfstest11 usage=dataOnly failureGroup=3,0,1 pool=datapool
%nsd: nsd=node11_data_sdg2  device=/dev/sdg servers=gpfstest11 usage=dataOnly failureGroup=3,0,1 pool=datapool
```

If any disks are previously used by GPFS, you must use the **-v no** flag to force GPFS to use them again.

**Note:** Use the **-v no** flag only if you are sure that the disk can be used by GPFS.
Use the # mmcrnsd -F diskfile [-v no] command to create NSDs and use the **mmlsnsd –m** command to display the NSDs.

## Apply GPFS FPO configuration changes

GPFS FPO requires several global GPFS configuration changes to operate successfully.

Set the GPFS page pool to 25% of system memory on each node. For Hadoop noSQL application, the page pool of GPFS FPO can be configured for better performance, for example, 30% of physical memory.

In this example, all nodes have the same amount of memory, which is a best practice. If some nodes have different memory, set the page pool on a per-node basis by using the **-N** flag.

```
# TOTAL_MEM=$(cat /proc/meminfo | grep MemTotal | tr -d \"[:alpha:]\" | tr -d\"[:punct:]\" | tr -d \"[:blank:]\")
# PAGE_POOL=$((${TOTAL_MEM}*25/(100*1024)))
# mmchconfig pagepool=${PAGE_POOL}M
```

Start the GPFS cluster:

```
# mmstartup -a
# mmgetstate -a
```

Use the **mmlsconfig** and **mmdiag** commands to see the configuration changes:

```
# mmlsconfig
# mmdiag –config
```

## Create the GPFS file system and pools

After you create NSDs, a GPFS file system can be created.

To use FPO, a single file system is recommended. The following example creates a file system with mount point **/mnt/gpfs** that is set to auto mount. This mount point is used in Hadoop configuration later. The replication for both data and metadata is set to 3 replicas. Quotas are not activated on this file system. An inode size of 4096 is recommended for typical **MapReduce**data sizes **-S** and **-E** settings help improve performance for **mtime** and **atime** updates. The **mmcrfs** command also creates GPFS storage pools based on the disk file %pools setting.

```
# mmcrfs gpfs-fpo-fs -F diskfile -T /mnt/gpfs -n 32 -m 3 -M 3 -r 3 -R 3 -i 4096 -A yes -Q no -S relatime -E no [-v no]
```

For more information on the pool configuration, see “Create GPFS Network Shared Disks (NSD)” on page 446.

Mount the file system on all nodes:

```
# mmmount all -a
```

Use the **mmlsfs** command to display the file system configuration:

```
# mmlsfs all
```

Use the **mmlsdisk** command to display the status of the NSDs:

```
# mmlsdisk gpfs-fpo-fs –L
```

Use the **mmdf** command to view the disk usage for the file system:

```
# mmdf gpfs-fpo-fs
```

Use the **mmlspool** command to view the storage pools:

```
# mmlspool gpfs-fpo-fs all –L
```

## Create GPFS Data Placement Policy

Before data can be written to a GPFS file system that has more than one pool, you must apply a data placement policy.

In this example, all of the data goes to data pool.

```
# cat policyfile
rule default SET POOL 'datapool'
```

After you create the rule file, use the **mmchpolicy** command to enable the policy:

```
# mmchpolicy gpfs-fpo-fs policyfile -I yes
```

Use the **mmlspolicy** command to display the currently active rule definition:

```
# mmlspolicy gpfs-fpo-fs –L
```

## Create file sets for MapReduce intermediate and temporary data

To efficiently store MapReduce intermediate and temporary data, use GPFS file sets and policies to better emulate local disk behavior.

**Note:** If **MapReduce** intermediate and temporary data is not stored on GPFS, **mapred.cluster.local.dir** in MRv1 or **yarn.nodemanager.log-dirs** and **yarn.nodemanager.local-dirs** in Hadoop Yarn does not point to a GPFS directory, you do not need to go through this section.

### Create an independent file set

Consider using **--inode-space new [--inode-limit MaxNumInodes[:NumInodesToPreallocate]** to create an independent file set. This can improve the performance for the file set but requires calculation for **MaxNumInodes** and **NumInodesToPreallocate**. **MaxNumInodes** must be eight times the number of files expected on the file set, and **NumInodesToPreallocate** must be half the value of **MaxNumInodes**. See the **mmcrfileset** man page to understand this option.

Use the **mmcrfileset** command to create two file sets, one for local intermediate data and one for temporary data:

```
# mmcrfileset gpfs-fpo-fs mapred-local-fileset
# mmcrfileset gpfs-fpo-fs mapred-tmp-fileset
```

After the file set is created, it must be linked to a directory under this GPFS file system mount point. This example uses **/mnt/gpfs/mapred/local** for intermediate data and **/mnt/gpfs/tmp** for temporary data. As **/mnt/gpfs/mapred/local** is a nested directory, the directory structure must exist before linking the file set. These two directories are required for configuring Hadoop.

```
# mkdir -p $(dirname /mnt/gpfs/mapred/local)
# mmlinkfileset gpfs-fpo-fs mapred-local-fileset -J /mnt/gpfs/mapred/local
# mmlinkfileset gpfs-fpo-fs mapred-tmp-fileset -J /mnt/gpfs/tmp
```

Use the **mmlsfileset** command to display file set information:

```
# mmlsfileset gpfs-fpo-fs -L
```

The next step to setting up the file sets is to apply a GPFS policy so the file sets act like local directories on each node. This policy instructs GPFS not to replicate the data for these two file sets, and since these file sets are stored on the data pool, they can use FPO features that keeps local writes on local disks. Metadata must still be replicated three times, which can result in performance overhead. File placement policies are evaluated in the order they are entered, so ensure that the policies for the file sets appear before the default rule.

```
# cat policyfile
rule 'R1' SET POOL 'datapool' REPLICATE (1,1) FOR FILESET ('mapred-local-fileset')
rule 'R2' SET POOL 'datapool' REPLICATE (1,1) FOR FILESET ('mapred-tmp-fileset')
rule default SET POOL 'datapool'
# mmchpolicy gpfs-fpo-fs policyfile -I yes
```

Use the **mmlspolicy** command to display the currently active rule definition:

```
# mmlspolicy gpfs-fpo-fs –L
```

In each of these file sets, create a subdirectory for each node that run Hadoop jobs. Based on the sample environment, this script creates these subdirectories:

```
# cat mk_gpfs_local_dirs.sh
#!/bin/sh for nodename in $(mmlsnode -N all); do
 mkdir -p /mnt/gpfs/tmp/${nodename}
 mkdir -p /mnt/gpfs/mapred/local/${nodename}
```

```
done
```

After that, on **${nodename}, link /mnt/gpfs/tmp/${nodename} /hadoop/tmp; link /mnt/gpfs/mapred/local/${nodename} /hadoop/local**. Then, in Hadoop cluster, configure **/hadoop/tmp** as **hadoop.tmp.dir** in all Hadoop nodes; configure **/hadoop/local** as **mapred.cluster.local.dir** in MRv1 or **yarn.nodemanager.log-dirs** and **yarn.nodemanager.local-dirs** in Hadoop Yarn for Hadoop nodes.

To check that the rules are working properly, you can write some test files and verify their replication settings. For example:

Create some files:

```
# echo "test" > /mnt/gpfs/mapred/local/testRep1
# echo "test" > /mnt/gpfs/testRep3
```

Use the **mmlsattr** command to check the replication settings

```
# mmlsattr /mnt/gpfs/mapred/local/testRep1
replication factors
metadata(max) data(max) file [flags]
 -----------------------------------
 1 ( 3) 1 ( 3) /mnt/gpfs/mapred/local/testRep1
 # mmlsattr /mnt/gpfs/testRep3
replication factors
metadata(max) data(max) file [flags]
-----------------------------------
3 ( 3) 3 ( 3) /mnt/gpfs/testRep3
```

### Set file system permissions

Depending on how different users interact with GPFS, you must create a user directory with permissions that allow users to create their own home directories.

```
# mkdir -p /mnt/gpfs/user
# chmod 1777 /mnt/gpfs/user
```

To make sure that MapReduce jobs can write to the GPFS file system, assign permissions to the CLUSTERADMIN user. CLUSTERADMIN is the user who starts Hadoop **namenode** and **datanode** service, for example, user hdfs.

```
# chown -R CLUSTERADMIN:CLUSTERADMINGROUP /mnt/gpfs
# chmod -R +rx /mnt/gpfs
```

Use the **ls** command to verify the permission settings:

```
# ls -lR /mnt/gpfs
```

## Tuning your operating system for IBM Spectrum Scale

Regardless of the Hadoop distribution being used, review the following information and take the necessary action to update your configuration:

For IBM AIX and Linux platform-specific considerations, see Chapter 3, “Configuring and tuning your system for GPFS,” on page 29.

To avoid any potential known issues, see the *Configuration and tuning questions* section in the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Ensure the IO scheduler for the disks used by GPFS is set to **deadline**. You can use the following command to change disk IO scheduler online for all disks used by GPFS:

```
echo "deadline" > /sys/block/<diskname>/queue/scheduler
```

If you want the change to survive reboot, you need to enter the command under the `/etc/rc.local` folder.

## Tuning IBM Spectrum Scale configuration for FPO

FPO-enabled clusters have a different architecture than traditional IBM Spectrum Scale deployments. Therefore, many of the default configuration parameters that are not suitable for FPO deployments must be modified.

Use the Table 39 as a guide to update your cluster configuration settings.

Use the **mmchconfig** and **mmlsconfig** commands to change and list the current value of any parameter. In the following example, **readReplicaPolicy** parameter is changed to the local setting. The example also shows how to specific multiple configuration parameters in a single command.

```
#shows current setting of readReplicaPolicy parameter
mmlsconfig |grep -i readReplicaPolicy
#changes the value to policy. Use -i ensure change immediate and persistent across node reboots.
 mmchconfig readReplicaPolicy=local -i
mmchconfig maxStatCache=100000,maxFilesToCache=100000
```

To replace the parameter name and values as required run these commands on one node in the cluster, and IBM Spectrum Scale propagates the changes to all other nodes.

**Note:** Some parameters such as **maxStartCache** and **maxFilesToCache** do not take effect until IBM Spectrum Scale is restarted. IBM Spectrum Scale can be restarted by using the following command.

```
/usr/lpp/mmfs/bin/mmumount all -a
/usr/lpp/mmfs/bin/mmshutdown -a
/usr/lpp/mmfs/bin/mmstartup -a
#Ensure GPFS daemon has started on all nodes in the GPFS cluster.
mmgetstate -a
```

You can specify multiple configuration parameters in a single **mmchconfig** command by using comma to separate each configuration. For example, the following command can be used to set most of the parameters specified in Table 40 on page 454:

```
mmchconfig
   readReplicaPolicy=local,restripeOnDiskFailure=yes,syncBuffsPerIteration=1,minMissedPingTimeout=60,
   leaseRecoveryWait=65,prefetchAggressivenessRead=2,prefetchAggressivenessWrite=0,
   maxFilesToCache=100000,maxStatCache=100000,worker1Threads=72,nsdMinWorkerThreads=48,
   nsdInlineWriteMax=1M,nsdSmallThreadRatio=2,nsdThreadsPerQueue=10,forceLogWriteOnFdatasync=no,
   disableInodeUpdateOnFdatasync=yes,unmountOnDiskFail=meta,dataDiskCacheProtectionMethod=2
```

*Table 39. IBM Spectrum Scale configuration parameter*

| Parameter Name | Default Value | New Value | Comment |
|---|---|---|---|
| **readReplicaPolicy** | Random | local | Enables the policy to read replicas from local disks. |
| **restripeOnDiskFailure** | No | Yes | Specifies whether IBM Spectrum Scale attempts to automatically recover from certain common disk failures. |

*Table 39. IBM Spectrum Scale configuration parameter (continued)*

| Parameter Name | Default Value | New Value | Comment |
|---|---|---|---|
| **metadataDiskWaitTimeForRecovery** | 2,400 seconds | <see comments to customize> | Sets delay period before start of recovery for failed metadata disks. Used when **RestripeOnDiskFailure** is set to yes. Ensure that the delay period is large enough to cover reboot time of the nodes hosting metadata disk servers. |
| **dataDiskWaitTimeForRecovery** | 3,600 seconds | <see comments to customize> | Sets delay period before start of recovery for failed data disks. Used when **RestripeOnDiskFailure** is set to yes.<br><br>Datadisks are controlled through a separate tunable due to their distribution across the variety of node types.<br><br>Ensure that the delay period is large enough to cover the reboot of the slowest node in the cluster. |
| **syncBuffsPerIteration** | 100 | 1 | Used to expedite buffer flush and the rename operations done by **MapReduce** jobs. |
| **minMissedPingTimeout** | 3 (seconds) | 10-60 (seconds) | Sets the lower bound on a missed ping timeout. For FPO clusters, a longer grace time is desirable before marking a node as dead, as it impacts all associated disks. Additionally, when running **MapReduce** workloads, the CPU can become overly busy and cause delayed ping responses. However, a longer timeout implies delay in recovery. A value between 10– 60 seconds is recommended. This value generally provides a good balance between the time to detect the real failures and the rate of false failure detection triggered by a delayed ping response due to CPU or network overload. |

*Table 39. IBM Spectrum Scale configuration parameter (continued)*

| Parameter Name | Default Value | New Value | Comment |
|---|---|---|---|
| **leaseRecoveryWait** | 35 | 65 | Allows a larger grace window before starting the recovery. |
| **pagepool** | Varied | 25% of | Sets the amount of physical memory that is reserved for cache on a node (use **-N** to list nodes that apply). |
| **prefetchPct** | 20(% of page pool) | See comments | Used by IBM Spectrum Scale as a guideline to limit the page pool space used for prefetch or write-behind buffers. For **MapReduce** workloads, sequential read and write, increase this parameter up to its 60% maximum. |
| **prefetchThreads** | 72 | See Comments | Controls the maximum possible threads that are dedicated to prefetching data for sequential file reads or to handle sequential write-behind.<br><br>Prefetch threads must have twice the number of disksavailable to the node. Default must work well in most configurations. |
| **maxFilesToCache** | 4,000 | 100,000 | Specifies the number of I nodes to cache.<br><br>Storing the inode of a file in cache permits faster reaccess to the file when retrieving location information for data blocks. Increasing this number can improve throughput for workloads with high file reuse and Hadoop **MapReduce** tasks. However, increasing this number excessively might cause paging at the file system manager node. The value must be large enough to handle the number of concurrently open files and allow caching of recently used files. |

*Table 40. IBM Spectrum Scale configuration parameter*

| Parameter Name | Default value | New value | Comment |
|---|---|---|---|
| **maxStatCache** | 1,000 | 512 | Specifies the number of I nodes to keep in the stat cache. The stat cache maintains only enough inode information to perform a query on the file system.<br><br>**Note:** The stat cache is not effective on the Linux operating system. Therefore, you need to set the **maxStatCache** attribute to a smaller value, such as 512, on that operating system. |
| **worker1Threads** | 48 | See comments | Controls the maximum number of concurrent file operations at any one instant, primarily for random read and write operations that cannot be prefetched. For big data applications, increase this setting to 72. |
| **nsdMinWorkerThreads** | 16 | 48 | Increases NSD server performance by providing many dedicated threads for NSD service. |
| **nsdInlineWriteMax** | 1,024 | 1,000,000 | Defines the amount of data sent inline with write requests to the NSD server; helps to reduce overhead caused by inter-node communication. |
| **nsdSmallThreadRatio** | 0 | 2 | Changing this parameter to 2 results in assigning more resources for large I/O for Hadoop workloads. |
| **nsdThreadsPerQueue** | 3 | 10 | Shows the number of threads servicing an internal I/O queue for disk operations.<br><br>Increasing this value results in increased parallelism for disk I/O and can increase throughput. |

*Table 40. IBM Spectrum Scale configuration parameter (continued)*

| Parameter Name | Default value | New value | Comment |
|---|---|---|---|
| **forceLogWriteOnFdatasync** | Yes | No | Controls forcing log writes to disk. When set to no, the IBM Spectrum Scale log record is flushed only when a new block is allocated to the file. |
| **disableInodeUpdateOnFdatasync** | No | Yes | When 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. |
| **unmountOnDiskFail** | No | meta | Controls how the IBM Spectrum Scale daemon responds when a disk failure is detected.<br><br>When set to **meta**, the file system is unmounted when metadata is no longer accessible. With replication factor set to 3, three failure groups must have at least one disk each in the failed state before a file system is unmounted |
| **cnfsReboot** | No | Yes | The **enforceFilesetQuotaOnRoot** parameter controls whether fileset quota must limit root user as any other users. The default value is No.<br><br>If this parameter is set, the fileset quota is enforced on root |

*Table 40. IBM Spectrum Scale configuration parameter (continued)*

| Parameter Name | Default value | New value | Comment |
|---|---|---|---|
| **dataDiskCacheProtectionMethod** | 0 | 2 | Defines the kind of disks used for the GPFS file system. The default 0 indicates that disks are **powerProtected** and no recovery is needed beyond standard GPFS log recovery.<br><br>A value of 2 ensures that when a node goes down, the data lost in the disk's cache is rebuilt by the GPFS.<br>**Note:** If the physical disk-write cache is enabled, upgrade IBM Spectrum Scale to 4.1.1.2 or later and set this configuration to 2 with auto recovery enabled. |

The IBM Spectrum Scale cluster and file system needs to be set up using the planning and best practices described in the "Configuring IBM Spectrum Scale Clusters" on page 446. IBM BigInsights® must be installed before configuring the IBM Spectrum Scale Hadoop connector and Hadoop applications. IBM Platform Symphony® can be installed after the IBM Spectrum Scale file system is created.

## Ingesting data into IBM Spectrum Scale clusters

`MapReduce` tasks perform best when input data is evenly distributed across cluster nodes. You can use the following approaches or a combination to ingest data for the first time and on an ongoing basis:

- Import data through a diskless IBM Spectrum Scale node. This ensures that the data is distributed evenly across all failure groups and all nodes within a failure group.
- If you have a large set of data to copy, it might help to use all cluster nodes to share ingest workload. Use a **write-affinity** depth of 0, along with as many cluster nodes with storage as possible to copy data in parallel.
- A **write-affinity** depth of 0 ensures that each node distributes data across as many nodes as possible. IBM Spectrum Scale policies can be used to enforce write-affinity depth settings based on fileset name, filename, or other attributes.
- Another mechanism to distribute data on ingest is to use **write-affinity depth failure** groups (WADFG) to control placement of the file replica. A WADFG setting of "*,*,*" ensures that all the file chunks are evenly distributed across all nodes. A placement policy can be used to selectively specify this attribute on the data set being ingested.

It is possible that even after employing the above techniques to ingest, the cluster might become unbalanced as nodes and disks are added or removed. You can check whether the data in the cluster is balanced by using the **mmdf** command. If data disks in different nodes are showing uneven disk usage, rebalance the cluster by running the **mmrestripefs -b** command. Keep in mind that the rebalancing command causes additional I/O activity in the cluster. Therefore, plan to run it at a time when workload is light.

## Exporting data out of IBM Spectrum Scale clusters

In many applications, it might be required to export the output data into another system or application for further use. Hadoop native components such as Flume can be used for this purpose. If HDFS Transparency is used for Hadoop applications, distcp feature is supported over IBM Spectrum Scale to export data into a remote HDFS file system or IBM Spectrum Scale file system. Additionally, since IBM Spectrum Scale provides POSIX semantics, custom scripts can be written to move data from an IBM Spectrum Scale cluster to any other POSIX-compliant file system. Consider using CNFS to copy the needed data directly.

If data must be exported into another IBM Spectrum Scale cluster, the AFM function can be used to replicate data into a remote IBM Spectrum Scale cluster.

## Upgrading FPO

When the application that runs over the cluster can be stopped, you can shut down the entire GPFS cluster and upgrade FPO. However, if the application over the cluster cannot be stopped, you need to take the rolling-upgrade procedure to upgrade nodes.

During this kind of upgrade, the service is interrupted. In production cluster, service interrupt is not accepted by the customers. If such cases, you need to take the rolling upgrade to upgrade node by node (or failure group by failure group) while keeping GPFS service up in the other nodes.

The guide for rolling upgrade is as follows:

- Only upgrade nodes from the same failure group at the same time slot; not operate nodes from two or more failure groups because bringing nodes from more than 1 failure groups will make your data exposed in data lost risk.
- Not break the quorum relationship when bringing down the nodes from one failure group. Before you bring down the nodes in one failure group, you need to check the quorum node. If bringing the quorum node in the to-be-operated failure group will break the quorum relationship in the cluster, you need to exclude that node for the rolling upgrade of the failure group.

**Prerequisites**

- Ensure that all disks are in a `ready` status and `up` availability. You can check by issuing the **mmlsdisk fs-name -L** command.
- Verify whether the upgraded-to GPFS version is compatible with the running version from IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html). For example, you cannot upgrade GPFS from 3.4.0.x directly into 3.5.0.24. You need to upgrade to 3.5.0.0 first and then upgrade to the latest PTF. You also need to verify whether the operating system kernel version and the Linux distro version are compatible with GPFS from IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- Find a time period when the whole system work load is low or reserve a maintenance time window to do the upgrade. When cluster manager or file system manager is down intentionally or accidentally, another node is elected to take the management role. But it takes time to keep the cluster configuration and the file system data consistent.
- When a file system manager is elected by cluster manager, it does not change even if the file system is unmounted in this node. If the file system is mounted in other nodes, it is also 'internal' mounted in the file system manager. This does not affect your ability to unload the kernel modules and upgrade GPFS without a reboot.

Upgrade FPO as follows:

1. Disable auto recovery for disk failure

To do a rolling upgrade of GPFS, you must shut down GPFS in some nodes. Disks in those nodes are unreachable during that time. It is better to handle this disk down manually with the following step.

Run **mmchconfig restripeOnDiskFailure=no -i** in any node in cluster to disable auto recovery for disk failure. It is necessary to include **-i** option for this change to take effect immediately and permanently. GPFS **restripeOnDiskFailure** is a cluster-wide configuration. So you need to run it only once in any one node in your cluster.

2. Get the list of nodes for this upgrade cycle

   Each upgrade cycle, you can only upgrade GPFS in nodes whose disks in it have the same first two numbers in failure group vector. Save node list in file **nodeList**, one node name per line in it. For general information on how to specify node names, see "Specifying nodes as input to GPFS commands" on page 67.

3. Get disk list in nodes for this upgrade cycle

   Get all disks attached in nodes for this upgrade cycle. Save it in file **diskList**. Each line in file **diskList** saves a disk name. **mmlsdisk –L** command can show which disks belong to the nodes you want to upgrade in this cycle.

4. Stop applications by using GPFS file system

   Confirming that there is no application which still opens file in GPFS file system. You can use **lsof** or **fuse** command to check whether there is still an open instance active for file in GPFS file system.

5. Unmount GPFS file system

   Unmount GPFS file system in all nodes for this upgrade cycle through command:

   ```
   mmumount <fsName> -N <nodeList>
   ```

   Confirming file system has already unmounted in all related nodes through command:

   ```
   mmlsmount <fsName> -L
   ```

6. Suspend disks

   Suspend all attached disks in nodes for this upgrade cycle to make sure that GPFS does not try to allocate new data block from these disks. GPFS can still read valid data block from suspended disk

   ```
   mmchdisk <fsName> suspend -d <diskList>
   ```

   Confirming disks are suspended properly through command:

   ```
   mmlsdisk <fsName>
   ```

7. Shut down GPFS

   Shut down GPFS in all nodes for this upgrade cycle through command:

   ```
   mmshutdown -N <nodeList>
   ```

   Confirming GPFS is in down status in these nodes through command:

   ```
   mmgetstate -a
   ```

8. Upgrade GPFS

   You can upgrade GPFS packages in each node of this upgrade cycle.

   For general information on how to install GPFS packages on nodes, see the following topics in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*:

   - *Installing IBM Spectrum Scale on Linux nodes and deploying protocols*
   - *Installing IBM Spectrum Scale on AIX nodes*
   - *Installing IBM Spectrum Scale on Windows nodes*

9. Start GPFS

   After all packages are ready, you can start up GPFS:

   ```
   mmstartup -N <nodeList>
   ```

   Confirming GPFS are in "Active" status in these nodes through command:

   ```
   mmgetstate -a
   ```

10. Resume and start disks

When GPFS is in "Active" status in all nodes, you can resume disks which were suspended intentionally in Step 7.

```
mmchdisk <fsName> resume -a or
mmchdisk <fsName> resume -d <diskList>
```

When these disks are in "ready" status and if some of these disks are in “down” availability, you can start these disks through the following command:

```
mmchdisk <fsName> start -a or
mmchdisk <fsName> start -d <diskList>
```

This might take a while since GPFS must do incremental data sync up to keep all data in these suspended disks are up-to-date. The time it needs depends on how much data has changed when the disks were kept in `suspende` status. You have to wait for **`mmchdisk start`** command to finish to do next step.

Confirming all disks are in `ready`status and `up` state through command:

```
mmlsdisk <fsName>
```

11. Mounts GPFS file system

    When all disks in the file system are in `up` status you can mount file system:

    ```
    mmmount <fsName> -N <nodeList>
    ```

    Confirming GPFS file system is mounted properly through command:

    ```
    mmlsmount <fsName> -L
    ```

    **Repeat Step 3 to Step 12 to upgrade GPFS in all nodes in your cluster.**

12. Enable auto recovery for disk failure

    Now you can enable auto recovery for disk failure through command:

    ```
    mmchconfig restripeOnDiskFailure=no -i
    ```

    It's necessary to includes **`-i`** option for this change to take effect immediately and permanently.

13. Upgrade GPFS cluster version and file system version

    Issue **`mmchconfig release=LATEST`** and **`mmchfs -V compat`** to ensure the upgrade is successful and the cluster would never revert to the old build for minor GPFS upgrade. It is recommended to use **`mmchfs –V compat`** to enable backward-compatible format changes.

    For major GPFS upgrade, consult IBM Support Center to verify compatibility between the different GPFS major versions, before issuing **`mmchfs –V full`**. For information about specific file system format and function changes when you upgrade to the current release, see Chapter 11, “File system format changes between versions of GPFS,” on page 109.

## Monitoring and administering IBM Spectrum Scale FPO clusters

IBM Spectrum Scale supports the use of the simple network management protocol (SNMP) for monitoring the status and configuration of the IBM Spectrum Scale cluster. By using an SNMP application, the system administrator can get a detailed view of the system and receive instant notification of important events, such as a node or disk failure.

The SNMP agent software consists of a master agent and a set of subagents, which communicate with the master agent through an agent/subagent protocol, the AgentX protocol in this case.

The SNMP subagent runs on a collector node of the IBM Spectrum Scale cluster. The collector node is designated by the system administrator by using the **`mmchnode`** command.

The Net-SNMP master agent, also called as the SNMP daemon, or `snmpd`, must be installed on the collector node to communicate with the IBM Spectrum Scale subagent and with your SNMP management application. Net-SNMP is included in most Linux distributions and must be supported by your Linux vendor.

For more information about enabling SNMP support, see the *GPFS SNMP support* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

Refer to the GPFS SNMP support topic in the *IBM Spectrum Scale: Administration Guide* for further information about enabling SNMP support.

**If using IBM BigInsights**

When you install IBM Spectrum Scale, you can enable IBM Spectrum Scale monitoring using the IBM BigInsights installation program. If the monitoring was not enabled at the time of installation, it can be done later by installing the Net-SNMP master agent on the collector node to communicate with the IBM Spectrum Scale subagent and the IBM BigInsights Console. Detailed instructions are provided in the Enabling monitoring for GPFS topic in the IBM InfoSphere® BigInsights Version 2.1.2 documentation.

**If using Platform Symphony or Hadoop distribution from other vendor**

You can leverage IBM Spectrum Scale SNMP integration for centralized monitoring of the IBM Spectrum Scale cluster. Follow the procedure outlined in the *GPFS SNMP support* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

# Rolling upgrades

During a regular upgrade, the IBM Spectrum Scale service is interrupted. For a regular upgrade, you must shut down the cluster and suspend the application workload of the cluster. During a rolling upgrade, there is no interruption in the IBM Spectrum Scale service. In a rolling upgrade, the system is upgraded node by node or failure group by failure group. During the upgrade, IBM Spectrum Scale runs on a subset of nodes.

In a rolling upgrade, nodes from the same failure group must be upgraded at the same time. If nodes from two or more failure groups stop functioning, only a single data copy is available online. Also, if the quorum node stops functioning, the quorum relationship in the cluster is broken. Therefore, the quorum node must be excluded from the rolling upgrade of the failure node.

## Performing a rolling upgrade

This topic lists the steps to perform a rolling upgrade.

- Ensure that the status of all disks is Ready and the availability is Up by running the **mmlsdisk <fs-name> -L** command.
- Verify the compatibility of the new IBM Spectrum Scale version with the running version by reviewing the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html). For example, IBM Spectrum Scale cannot be upgraded from 3.4.0.x to 3.5.0.24 before being upgraded to 3.5.0.0.
- Verify the compatibility of the planned upgrade system kernel and Linux distro versions with IBM Spectrum Scale by reviewing the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).
- While performing maintenance on the cluster manager and the file system manager nodes, the nodes fail over automatically. However, you must manually assign the cluster manager and the file system manager to other nodes by using the **mmchmgr** command when the cluster is not busy.

1. Disable auto recovery for disk failure.

   To upgrade a node, shut down IBM Spectrum Scale running on the node. When IBM Spectrum Scale is shut down, disks in the node cannot be reached. Instead of letting the disks fail and the automatic recovery initiate, temporarily disable auto recovery.

   Run the **mmchconfig restripeOnDiskFailure=no -i** command to disable auto recovery for disk failure. With the **-i** option, the parameter takes effect immediately and permanently. For example, in small clusters, the node number is less than 30 nodes. Therefore, it takes a shorter time for IBM

Spectrum Scale to synchronize the configuration. For large clusters, the node number is in hundreds. Therefore, the time taken to synchronize the configuration is longer. The **restripeOnDiskFailure** parameter is a cluster-wide configuration.

After disabling auto recovery, check for auto recovery in the file system manager by running the following commands:

- If there are multiple file systems in the cluster, run **mmlsmgr** command to check the fs manager of a single file system.
- Log in to the fs manager of the file system and run **ps -elf | grep -e tschdisk -e tsrestripefs** command. If there are processes running, wait for them to complete.

2. Select the nodes that must be upgraded and schedule the time of each upgrade. In each upgrade cycle, you can only upgrade IBM Spectrum Scale on nodes where the disks have the same first two numbers in the failure group. Save the list of nodes in the nodeList file with one node name on each line. Save a list of the disks on the nodes that will be upgraded in this cycle in the diskList file, with each line containing an NSD name. Run the **mmlsdisk Device –M** command to check which disks belongs to which node.
3. Stop all applications that are using the IBM Spectrum Scale file system before stopping IBM Spectrum Scale. To check for open files in the file system, run the **lsof** or the **fuse** command.
4. Unmount the IBM Spectrum Scale file system on all nodes by running the following command: **mmumount <fsName> -N <nodeList>**

   To confirm that the file system has been umounted on all related nodes, run the following command: **mmlsmount <fsName> -L**
5. Suspend all disks in the nodes so that IBM Spectrum Scale does not allocate new data blocks from these disks. IBM Spectrum Scale can still read data block from suspended disks by running the following command:

   **mmchdisk <fsName> suspend -d <diskList>**

   To confirm that all disks are suspended properly, run the following command: **mmlsdisk <fsName>**
6. Shut down IBM Spectrum Scale on the nodes by running the following command:

   **mmshutdown -N <nodeList>**

   To confirm IBM Spectrum Scale has stopped functioning on these nodes, run the following command: **mmgetstate -a**

   Upgrade IBM Spectrum Scale packages on each node. For information on how to install IBM Spectrum Scale packages on node, see the following topics:

   - *Installing IBM Spectrum Scale on Linux nodes and deploying protocols* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
   - *Installing IBM Spectrum Scale on AIX nodes* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*
   - *Installing IBM Spectrum Scale on Windows nodes* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*

   After everything has been installed and the portability layer has been built, start IBM Spectrum Scale by running the following command: **mmstartup -N <nodeList>**

   To confirm that IBM Spectrum Scale is active on the upgraded nodes, run the following command: **mmgetstate -a**.

   Resume all the suspended disks by running the following commands: **mmchdisk <fsName> resume -a** or **or mmchdisk <fsName> resume -d <diskList>**.

   If some of the suspended disks are in the Down availability, start these disks by running the following command: **mmchdisk <fsName> start -a** or **mmchdisk <fsName> start -d <diskList>**.

   This may take a while because IBM Spectrum Scale is performing an incremental data sync up to keep the data in these suspended disks up-to-date. The time taken depends on the data that has been changed while the disks were kept in the Suspended status. Wait for the **mmchdisk <fsName> start [. . .]** command to finish before moving on to the next step.

To confirm that all disks are in the ready state, run the following command: **mmlsdisk <fsName>**.

7. When all the disks in the file system are functioning, mount the file system by running the following command: **mmmount <fsName> -N <nodeList>**

   Confirm that the IBM Spectrum Scale file system has mounted by running the following command: **mmlsmount <fsName> -L**

8. Perform Step through Step to upgrade IBM Spectrum Scale on all nodes in the cluster.
9. To enable auto recovery for disk failure, run the following command: **mmchconfig restripeOnDiskFailure=yes -i**

   Ensure that you use the **-i** option so that this change takes effect immediately and permanently.

10. Upgrade the IBM Spectrum Scale cluster version and file system version

    If all applications run without any issues, run the **mmchconfig release=LATEST** command to upgrade the cluster version to the latest. Then, run the **mmchfs -V compat** command to ensure that the upgrade is successful. To enable backward-compatible format changes, run **mmchfs -V compat**.

    **Note:** After running the **mmchconfig release=LATEST** command, you cannot revert the cluster release version to an older version. After running the **mmchfs -V compat** command, you cannot revert the file system version to an older version.

    For major IBM Spectrum Scale upgrade, check IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) or contact scale@us.ibm.com before running the **mmchfs –V full** command to verify the compatibility between the different IBM Spectrum Scale major versions. For information about specific file system format and function changes, see Chapter 11, "File system format changes between versions of GPFS," on page 109.

### Upgrading other infrastructure

The same process of choosing nodes should be used when upgrading hardware firmware, operation system kernel or other components that require you to take IBM Spectrum Scale down on the node.

## The IBM Spectrum Scale FPO cluster

When a node reboots due to hardware or software issues, IBM Spectrum Scale can be started and the file system can be mounted if autoload is configured as yes in the mmchconfig command. In a typical FPO deployment, each node has several local attached disks. When a node stops functioning, disks attached to the node are made unavailable from the cluster.

**Note:** Do not reboot a node if the file system is still mounted.

After the node is rebooted, the disk status of the node is uncertain. The status of the node is dependent upon the auto recovery configuration (**mmlsconfig restripeOnDiskFailure**) and the IO operations over the cluster.

### Restarting a large IBM Spectrum Scale cluster

A cluster might have to be restarted because of an OS upgrade. On large FPO clusters auto-recovery must be disabled before restarting IBM Spectrum Scale.

- Ensure that the status of all disks is Ready and the availability is Up by running the **mmlsdisk <fs-name> -L** command.
- Verify the compatibility of the planned upgrade system kernel and Linux distro versions with IBM Spectrum Scale by reviewing the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

1. Disable auto recovery for disk failure.

   When IBM Spectrum Scale stops functioning, some nodes might not shut down. This might bring some disks down from the fast nodes and might trigger auto recovery. To avoid this, temporarily disable auto recovery.

Run the **mmchconfig restripeOnDiskFailure=no -i** command to disable auto recovery for disk failure. With the `-i` option, the parameter takes effect immediately and permanently. For example, in small clusters, the node number is less than 30 nodes. Therefore, it takes a shorter time for IBM Spectrum Scale to synchronize the configuration. For large clusters, the node number is in hundreds. Therefore, the time taken to synchronize the configuration is longer. The **restripeOnDiskFailure** parameter is a cluster-wide configuration.

After disabling auto recovery, check for auto recovery in the file system manager by running the following commands:

- If there are multiple file systems in the cluster, run **mmlsmgr** command to check the fs manager of a single file system.
- Log in to the fs manager of the file system and run **ps -elf | grep -e tschdisk -e tsrestripefs** command. If there are processes running, wait for them to complete.

2. Stop all applications that are using the IBM Spectrum Scale file system. To check for open files in the file system, run the **lsof** or the **fuse** command.

   For example, to check if IBM Spectrum Scale file system has processes using it run the following commands: **lsof +f -- /dev/name_of_SpectrumScale_filesystem** or **fuser -m /mount_point_of_SpectrumScale_filesystem**

3. Unmount the IBM Spectrum Scale file system on all nodes for this upgrade cycle by running the following command: **mmumount <fsName> -a**

   To confirm that the file system has been umounted on all related nodes, run the following command: **mmlsmount <fsName> -L**

4. To disable the Automatic mount option, run the following command: **mmchfs <fsName> -A no**

   **Note:** In a large cluster, some nodes might take a while to start. If the `-A` option is not set to **no**, unnecessary disk IO might cause some disks from slow nodes to be marked as non functional.

5. Shut down IBM Spectrum Scale on the nodes by running the following command:

   **mmshutdown -N <nodeList>**

   To confirm IBM Spectrum Scale has stopped functioning on these nodes, run the following command: **mmgetstate -a**

6. Upgrade IBM Spectrum Scale or perform the maintenance procedure on the whole cluster.

7. Start IBM Spectrum Scale cluster. After everything has been installed and the portability layer has been built, start IBM Spectrum Scale by running the following command: **mmgetstate -a**.

   To confirm that IBM Spectrum Scale is active on the upgraded nodes, run the following command: **mmgetstate -a**.

8. When IBM Spectrum Scale is active on all nodes, check the state of all disks by running the following command: **mmlsdisk <fsName> -e**. If some disks in the file system do not have the Up availability and the Ready status, run the **mmchdisk <fsName> start -a** command so that the disks start functioning. Run the **mmchdisk <fsName> resume -a** command so that the suspended and to-be-emptied disks become available.

9. When all the disks in the file system are functioning, mount the file system by running the following command: **mmmount <fsName> -N <nodeList>**

   Confirm that the IBM Spectrum Scale file system has mounted by running the following command: **mmlsmount <fsName> -L**

10. To enable auto recovery for disk failure, run the following command: **mmchconfig restripeOnDiskFailure=yes -i**

    Ensure that you use the `-i` option so that this change takes effect immediately and permanently.

11. To enable the Automatic mount option, run the following command: **mmchfs <fsName> -A yes**.

12. If you have upgraded IBM Spectrum Scale version in step 6, upgrade the IBM Spectrum Scale cluster version and file system version.

If all applications run without any issues, run the **mmchconfig release=LATEST** command to upgrade the cluster version to the latest. Then, run the **mmchfs -V compat** command to ensure that the upgrade is successful. To enable backward-compatible format changes, run **mmchfs -V compat**.

**Note:** After running the **mmchconfig release=LATEST** command, you cannot revert the cluster release version to an older version. After running the **mmchfs -V compat** command, you cannot revert the file system version to an older version.

For major IBM Spectrum Scale upgrade, check IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) or contact scale@us.ibm.com before running the **mmchfs -V full** command to verify the compatibility between the different IBM Spectrum Scale major versions. For information about specific file system format and function changes, see Chapter 11, "File system format changes between versions of GPFS," on page 109.

# Failure detection

## The node state

This topic describes how to check the node state in a IBM Spectrum Scale cluster.

To check the state of the nodes in a IBM Spectrum Scale cluster, run **mmgetstate** or **mmgetstate –a** command. Nodes with the Down status are reachable but the IBM Spectrum Scale daemon on those nodes is not functioning. Nodes with the Unknown status cannot be reached from the node on which **mmgetstate** command was run.

To follow-up on investigating the state of a node, check if the node is functioning or has a network issue.

## The disk state

This topic describes how to check the disk state in a IBM Spectrum Scale cluster.

To check the state of the disks in a IBM Spectrum Scale cluster, run **mmlsdisk -e** command. This command lists all the disks that do not have the Available or Up status.

## IBM Spectrum Scale log

This topic describes the IBM Spectrum Scale log files.

The IBM Spectrum Scale log files are saved in the `/var/adm/ras/` directory on each node. Each time the IBM Spectrum Scale daemon starts, a new log file is created. The mmfs.log.latest log file is the link to the latest log. On Linux, all additional information is sent to the system log in `/var/log/messages`.

Because the IBM Spectrum Scale cluster manager and file system managers handle cluster issues such as node leaves or disk down events, monitor the IBM Spectrum Scale log on the cluster manager and file system manager to get the best view of the cluster and file system status.

# Disk Failures

This section describes how to handle a disk failure.

In an FPO deployment model with IBM Spectrum Scale the **restripeOnDiskFailure=yes** configuration parameter should be set to `yes`. When a disk is not functioning, auto recovery enables the disk to start functioning. Auto recovery enlists the help of any node in the cluster to help recover data. This may affect the file system I/O performance on all nodes, because data might have to be copied from a valid disk to recover the disk.

## Stopping the disk failure auto recovery operation

The auto recovery operation can impact the IO performance across the cluster. Therefore, you can stop auto recovery manually and restart it when the cluster is not so busy. The disks that are not functioning must be recovered to protect your data.

Run the **mmlsdisk -e** command to see the disks that do not have the Up availability and the Ready status. If all the disks in the file system are functioning correctly, the system displays the following message: `6027-623 All disks up and ready.`

1. To stop the auto recovery process, stop the **tschdisk** and **tsrestripefs** processes on the file system manager node. Log in to the IBM Spectrum Scale file system manager node. Retrieve the **tschdisk** and **tsrestripefs** command processor ID through the **ps -elf | grep -e tschdisk -e tsrestripefs** command.

   Alternatively, check the IBM Spectrum Scale log (`/var/adm/ras/mmfs.log.latest`) in the file system manager node to see if there is **tschdisk** command is still running. When the **restripefs** command is invoked by the auto recovery and is still running, the command log message is redirected to `/var/adm/ras/restripefsOnDiskFailure.log.<timestamp>`(IBM Spectrum Scale 4.1 and IBM Spectrum Scale 4.1.1) or `/var/adm/ras/autorecovery.log.<timestamp>`(IBM Spectrum Scale 4.1.1 PTF1 and later).
2. Stop the **tschdisk** and **tsrestripefs** command processes by running the **kill -9 <pid>** command. Double check whether **tschdisk** and **tsrestripefs** command processes have been stopped by running the **ps -elf | grep -e tschdisk -e tsrestripefs** command.

## Starting the disk failure recovery

This topic lists the steps to start the disk failure recovery.

1. To check the disks that are not in the Ready state, run the **mmlsdisk -e** command.
2. Resume the disks that have the Suspended status.

   If there are multiple suspended disks, create a file that lists all the suspended disks one nsdname per line before you resume the disks. Resume the suspended disks by issuing the following command:

   ```
   mmchdisk <fsName> resume -d <suspendDisk:List>
   ```

   Check the disk status again by running the **mmlsdisk -e** command and confirm that all disks are now in the Ready state. If some disks are still in the Suspended state, there might be a hardware media or a connection problem. Save the names of these disks in the brokenDiskList file.
3. Save the disks that do not have the Up availability in the downDiskList file. Each line in downDiskList file stores a disk name. Start these disks by running the following command:

   ```
   mmchdisk <fsName> start -d <downDiskList>
   ```

   Check the disk status by running the **mmlsdisk -e** command to confirm that all disks have the Up availability. Disks that do not have the Up availability might have a hardware media or connection problem. Save the names of these disks in the tobeSuspendDiskList file. Suspend these disks by running the following command:

   ```
   mmchdisk <fsName> suspend -d <tobeSuspendDiskList>
   ```
4. If a disk is in Suspended status after you restart it, there might be a hardware media or connection problems. To keep your data safe, migrate it to the suspended disks by running the following command:

   ```
   mmrestripefs <fsName> -r
   ```

   After a file system restripe, all data in the suspended disks is migrated to other disks. At this point, all the data in the file system has the desired level of protection.
5. Check the disk connections and the disk media for disks that are in the Suspended state and repeat step 2 through step 4. If a failure occurs again, delete these disks from file system by running the **mmdeldisk** command. For example,

   ```
   mmdeldisk <fs-name> "broken-disk1;broken-disk2"
   ```

   If you are unable to delete a broken disk, contact IBM support.

### Handling physically broken disk

If a disk is physically broken, it cannot be recovered by auto recovery or manual recovery. Do not keep broken disks in the file system and schedule to delete them from the file system.

#### Deleting disks when auto recovery is not enabled (check this by `mmlsconfig restripeOnDiskFailure`):

Deleting NSD disks from the file system can trigger disk or network traffic because of data protection. If your cluster is busy with application IO and the application IO performance is important, schedule a maintenance window to delete these broken disks from your file system. Follow the steps in the "Starting the disk failure recovery" on page 465 section to check if a disk is physically broken and handle the broken disks.

#### Deleting disks when auto recovery is enabled (check this by `mmlsconfig restripeOnDiskFailure`):

When the IO operation is being performed on the physically broken disks, IBM Spectrum Scale marks the disks as non functional. Auto recovery suspends the disks if it fails to change the availability of the disk to Up and restripes the data off the suspended disks. If you are using IBM Spectrum Scale 4.1.0.4 or earlier, deleting the non functional disks triggers heavy IO traffic (especially for metadata disks). On IBM Spectrum Scale 4.1.0.4, **mmdeldisk** command has been improved. If the data on non functional disks have been restriped, the disk status will be Emptied. The **mmdeldisk** command deletes the non functional disks with the Emptied status without involving additional IO traffic.

## Node failure

In an FPO deployment, each node has locally attached disks. When a node fails or has a connection problem with other nodes in a cluster, disks in this node become unavailable. Reboot a node to repair a hardware issue or patch the operating system kernel. Both these cases are node failures.

If auto recovery is enabled, that is, the **restripeOnDiskFailure=yes** parameter is set to yes, and a failed node is recovered within auto recovery wait time (check the details described in the Auto Recovery for Disk Failure section), auto recovery handles the node failure automatically by bringing up down disks and ensuring all data has the desired replication. If a node is not recovered within the auto recovery wait time, auto recovery migrates the data off the disks in the failed node to other disks in cluster.

### Reboot node intentionally

#### Automatic recovery of a node

If you want to reboot a node or enable some configuration change that requires a reboot and have it recovered without auto recovery, check the auto recovery wait time. The auto recovery wait time is defined by the minimum value of minDiskWaitTimeForRecovery, metadataDiskWaitTimeForRecovery and dataDiskWaitTimeForRecovery. By default, minDiskWaitTimeForRecovery is 1800 seconds, metadataDiskWaitTimeForRecovery is 2400 seconds and dataDiskWaitTimeForRecovery is 3600 seconds. If the reboot is completed within the auto recovery wait time, it is safe to unmount the file system, shut down IBM Spectrum Scale, and reboot your node without having to disable auto recovery.

#### Manual recovery of a node

When you want to perform hardware maintenance for a node that must be shut down for a long time, follow the same steps mentioned in IBM Spectrum Scale Rolling Upgrade Procedure and perform hardware maintenance.

## Node crash and boot up

This topic lists the steps to recover a node automatically or manually.

**Recovering a node automatically:**

When a node crashes due to kernel or other critical issues and is recovered within the auto recovery wait period, IBM Spectrum Scale cluster manager can add this node automatically and IBM Spectrum Scale auto recovery brings up disks and repairs the dirty data in disks attached in the node. Check if the node and the disks in the node work normally and the data in the disks is updated.

1. Check whether IBM Spectrum Scale state is active on all nodes in the cluster by running the **mmgetstate –a** command. If a node is functional but IBM Spectrum Scale is not active, check IBM Spectrum Scale log (`/var/adm/ras/mmfs.log.latest`) on the node and run the **mmstartup** command after the issue in the log has been resolved.
2. Check if any disk does not have the Up availability and the Ready state by running the **mmlsdisk -e** command. If all disks in the file system are in the Ready state, the system displays the following message `6027-623 All disks up and ready`. Perform the disk recovery operation to change the state of all disks to Ready.
3. Run the **mmlsdisk** command to confirm that there is no warning message at the end of output.

   If you see the following message, there are data replicas on suspended and to-be-emptied disks. If the suspended and to-be-emptied disks are not physically broken, recover them and run the **mmrestripefs -r** command to fix the warning message. If the disks are physically broken, suspend them and run the **mmrestripefs –r** command to fix the warning message.

   **Attention:** Due to an earlier configuration change the file system may contain data that is at risk of being lost.

**Recovering a node manually:**

When a node crashes and is not recovered while auto recovery is temporarily suspended, start the node and recover the disks. In this case, IBM Spectrum Scale auto recovery migrates all data from disks in this node to other valid disks in cluster.

1. Find the root cause of the node crash, fix it, and recover the node.
2. If IBM Spectrum Scale autoload configuration is disabled, start the IBM Spectrum Scale daemon by running the **mmstartup** command. Check if the node state is Active. If the state is not active after a few minutes, check IBM Spectrum Scale log (`/var/adm/ras/mmfs.log.latest`) on this node and run the **mmstartup** command after fixing the issue.
3. When IBM Spectrum Scale is active, auto recovery is invoked automatically to recover the disks in this node. Check the IBM Spectrum Scale log (`/var/adm/ras/mmfs.log.latest`) and the restripefs log (`/var/adm/ras/restripefsOnDiskFailure.log.latest`) on the file system manager node for more details.

## Handling node crashes

If a failed node cannot be recovered, auto recovery migrates all data from disks in this node to other disks in cluster. If the system does not recover, delete the disks in the node and node.

1. Log in to another cluster node and run **mmlsdisk <fs-name> -M** command to get a list of disks attached to the failed node. Save the disk list in the diskList file. Each line lists a disk name.
2. Run the **mmdeldisk <fsName> -F <diskList>** command to delete the disks attached to the failed node.
3. Run the **mmdelnsd -F <diskList>** command to delete NSDs attached to the failed node. Run **mmdelnode** command to remove the node, or if you are replacing the node with new hardware, use the same name and IP address to continue.

   To replace the failed node with a new node, start the replacement mode with the hostname and the IP address of the failed node. Install IBM Spectrum Scale packages and configure SSH authorization with other nodes in the cluster. Run the following command to restore IBM Spectrum Scale configurations in this replacement node:

   ```
   mmsdrrestore -p <cluster manager>  -R <remoteFileCopyCommand> -N <replacement node>
   ```

Use the **mmlsmgr** command to identify the cluster manager node. Use the Remote file copy command that is configured for the cluster.

4. Start IBM Spectrum Scale on the replacement node by running the **mmstartup -N <replacement node>** command. Confirm that IBM Spectrum Scale state is active by running the **mmgetstate -N <replacement node>** command.
5. Prepare a stanza file to create NSDs by running the **mmcrnsd** command and add these disks into file system by running the **mmadddisk** command.

# Handling multiple nodes failure

Auto recovery must be enabled in an FPO cluster to protect data from multiple node failures. Set **mmchconfig restripeOnDiskFailure=yes -N all**.

Usually, if the concurrent failed nodes are less than maxFailedNodesForRecovery, auto recovery will protect data against node failure or disk failure. If the concurrent failed nodes are larger than maxFailedNodesForRecovery, auto recovery exits without any action and the administrator has to take some actions to recover it.

## Multiple nodes failure without SGPanic

This topic lists the steps to handle multiple nodes failure without SGPanic

1. Recover the failed nodes.
2. If all nodes are recovered quickly, run the **mmlsdisk <fs-name> -e** command to view the down disk list.
3. Run the **mmlsnsd -X** command to check whether there are disks that are undetected by the operating system of nodes. For example,

```
# mmlsnsd -X

Disk name    NSD volume ID      Device         Devtype  Node name                Remarks
---------------------------------------------------------------------------------------------
mucxs131d01  AC170E46561E7A8F   /dev/sdb       generic  mucxs131.muc.infineon.com server node
mucxs131d02  AC170E46561E7A90   /dev/sdc       generic  mucxs131.muc.infineon.com server node
mucxs531d07  AC170E4B5612838E   /dev/sdh       generic  mucxs531.muc.infineon.com server node
mucxs531d08  AC170E4B56128391   -              -        mucxs531.muc.infineon.com (not found) server node
```

   In the above output means the physical disk for the nsd mucxs531d08 is not recognized by the OS. If a disk is not detected, check the corresponding node to see if the disk is physically broken. If the undetected disks cannot be recovered quickly, remove them from the down disk list.
4. Run the **mmchdisk <fs-name> start -d <down disk in step3>**. If it succeeds, go to step5); if not, open PMR against the issue.
5. If the undetected disks cannot be recovered, run the **mmrestripefs <fs-name> -r** to fix the replica of the data whose part of replica are located in these undetected disks.

## SGPanic for handling node failure

For internal disk rick storage (FPO clusters), unmountOnDiskFail must be configured as “meta”. If it is not, change the configuration by running the **mmchconfig** command.

With unmountOnDiskFail configured as meta, if you see file system SGPanic reported when nodes are non functional, there are more than three nodes with metadata disk down together or there are more than three disks with meta data down. Follow the steps in the section 8.1 to fix the issues. Run the **mmfsck -n** command to scan the file system to ensure that mmfsck displays the following message: `File system is clean finally`. If mmfsck -n does not report “File system is clean”, you need to open PMR to report the issues and fix this with guide from IBM Spectrum Scale.

# Network switch failure

This topic describes how to handle network switch failure.

In an FPO cluster, if Auto recovery is enabled and there are more than maxFailedNodesForRecovery non functional nodes, auto recovery does not recover the nodes. By default, maxFailedNodesForRecovery is three nodes. You can change this number depending on your cluster configuration.

A switch network failure can cause nodes to be reported as non functional. If you want auto recovery to protect against switch network failures, careful planning is required in setting up the FPO cluster. For example, a network switch failure must not bring disks (with metadata) down from 3 or more failure groups, and maxFailedNodesForRecovery must be configured to a value that is larger than the number of down nodes that will result from a switch network failure.

## Data locality restore

In an FPO cluster, if the data storage pool is enabled with **allowWriteAffinity=yes**, the data locality is decided by the following order:

- WADFG is set by **mmchattr** or the policy.
- Default WAD or WAD is set by policy and the data ingesting node.

If the file is set with WADFG, the locality complies with WADFG independent of where the data is ingested. If the file is not set with WADFG, the locality is decided according to the WAD and data-ingesting node. Also, data locality configurations are the required configurations. If there are no disks available to comply with the configured data locality, the IBM Spectrum Scale FPO stores the data in other disks.

The data locality might be broken if there are **mmrestripefs -r** and **mmrestripefile** after disk failure or node failure. If your applications need data locality for good performance, restore the data locality after node failure or disk failure.

### Data locality impacted from down disks

All disks in a node must be configured as the same failure or locality group. After a disk is non functional, **mmrestripefs -r** from auto recovery suspends the disk and restripes the data on the non functional disks onto other disks in the same locality group. The data locality is not broken because the data from local disks is still in that node. If you do not have other disks available in the same locality group, **mmrestripefs –r** from auto recovery restripes the data on the non functional disks onto other nodes, breaking the data locality for the applications running over that node.

### Data locality impacted from the non functional nodes

If a non functional node does not have NSD disks in the file system, the data locality is not impacted. If the non functional node has NSD disks in the file system and the node is not recovered within dataDiskWaitTimeForRecovery (if all down disks are dataOnly disks) and metadataDiskWaitTimeForRecovery (if there is meta data NSD disk down), auto recovery suspends the disks and performs **restripefs -r**. All disks from the non functional nodes are not available for write and the data from the non functional disks is restriped onto other nodes. Therefore, the data locality is broken on the non functional nodes.

### Data locality impacted from unintended `mmrestripefile -b` or `mmrestripefs -b`

If the file is not set with WADFG (by policy or by mmchattr), both **mmrestripefile -b** and **mmrestripefs -b** might break the data locality.

### Data Locality impacted from unintended `mmrestripefile -l`

If the file is not set with WADFG (by policy or by mmchattr), **mmrestripefs -l** might break the data locality. The node running **mmrestripefile -l** is considered as the data writing node and all first replica

of data is stored in the data writing node for an FPO-enabled storage pool.

The following sections describe the steps to check if your data locality is broken and how fix it if needed.

## Check the data locality

This topic describes the steps to check the data locality.

```
cd /usr/lpp/mmfs/samples/fpo/
g++ -g -DGPFS_SNC_FILEMAP -o tsGetDataBlk -I/usr/lpp/mmfs/include/ tsGetDataBlk.C -L/usr/lpp/mmfs/lib/ -lgpfs
./tsGetDataBlk <filename>  -s 0 -f <data-pool-block-size * blockGroupFactor> -r 3
```

Check the output of the program tsGetDataBlk:

```
[root@gpfstest2 sncfs]# /usr/lpp/mmfs/samples/fpo/tsGetDataBlk /sncfs/test -r 3
File length: 1073741824, Block Size: 2097152
Parameters: startoffset:0, skipfactor: META_BLOCK, length: 1073741824, replicas 3
numReplicasReturned: 3, numBlksReturned: 4, META_BLOCK size: 268435456
Block 0 (offset 0) is located at disks:  2   4   6
Block 1 (offset 268435456) is located at disks:  2   4   6
Block 2 (offset 536870912) is located at disks:  2   4   6
Block 3 (offset 805306368) is located at disks:  2   4   6
```

In the above example, the block size of data pool is 2Mbytes, the blockGroupFactor of the data pool is 128. So, the META_BLOCK (or chunk) size is 2MB * 128 = 256Mbytes. Each output line represents one chunk. For example, Block 0 in the above is located in the disks with disk id 2, 4 and 6 for 3 replica.

In order to know the node on which the three replicas of Block 0 are located, check the mapping between disk ID and nodes:

Check the mapping between disks and nodes by **mmlsdisk** (the 9th column is the disk id of NSD) and **mmlsnsd**:

```
 [root@gpfstest2 sncfs]# mmlsdisk sncfs –L
disk         driver   sector     failure holds    holds                                storage
name         type       size       group metadata data  status        availability disk id pool         remarks
------------ -------- ------ ----------- -------- ----- ------------- ------------ ------- ------------ ---------
node1_sdb    nsd         512           1    Yes      No    ready         up                1 system        desc
node1_sdc    nsd         512       1,0,1    No       Yes   ready         up                2 datapool
node2_sda    nsd         512           1    Yes      No    ready         up                3 system
node2_sdb    nsd         512       2,0,1    No       Yes   ready         up                4 datapool
node6_sdb    nsd         512           2    Yes      No    ready         up                5 system        desc
node6_sdc    nsd         512       3,0,1    No       Yes   ready         up                6 datapool
node7_sdb    nsd         512           2    Yes      No    ready         up                7 system
node7_sdd    nsd         512       4,0,2    No       Yes   ready         up                8 datapool
node11_sdb   nsd        512           3    Yes      No    ready         up                9 system        desc
node11_sdd   nsd        512       1,1,1    No       Yes   ready         up               10 datapool      desc
node9_sdb    nsd         512           3    Yes      No    ready         up               11 system
node9_sdd    nsd         512       2,1,1    No       Yes   ready         up               12 datapool
node10_sdc   nsd        512           4    Yes      No    ready         up               13 system        desc
node10_sdf   nsd         512       3,1,1    No       Yes   ready         up               14 datapool
node12_sda   nsd        512           4    Yes      No    ready         up               15 system
node12_sdb   nsd        512       4,1,2    No       Yes   ready         up               16 datapool
[root@gpfstest2 sncfs]# mmlsnsd
 File system   Disk name    NSD servers
---------------------------------------------------------------------------
 sncfs         node1_sdb    gpfstest1.cn.ibm.com
 sncfs         node1_sdc    gpfstest1.cn.ibm.com
 sncfs         node2_sda    gpfstest2.cn.ibm.com
 sncfs         node2_sdb    gpfstest2.cn.ibm.com
 sncfs         node6_sdb    gpfstest6.cn.ibm.com
 sncfs         node6_sdc    gpfstest6.cn.ibm.com
 sncfs         node7_sdb    gpfstest7.cn.ibm.com
 sncfs         node7_sdd    gpfstest7.cn.ibm.com
 sncfs         node11_sdb   gpfstest11.cn.ibm.com
 sncfs         node11_sdd   gpfstest11.cn.ibm.com
 sncfs         node9_sdb    gpfstest9.cn.ibm.com
```

```
sncfs         node9_sdd    gpfstest9.cn.ibm.com
sncfs         node10_sdc   gpfstest10.cn.ibm.com
sncfs         node10_sdf   gpfstest10.cn.ibm.com
sncfs         node12_sda   gpfstest12.cn.ibm.com
sncfs         node12_sdb   gpfstest12.cn.ibm.com
```

The three replicas of Block 0 are located in disk id 2 (NSD name node1_sdc, node name is gpfstest1.cn.ibm.com), disk id 4 (NSD name node2_sdb, node name is gpfstest2.cn.ibm.com), and disk id 6 (NSD name node6_sdc, node name is gpfstest6.cn.ibm.com). Check each block of the file to see if the blocks are located correctly. If all blocks are not located correctly, fix the data locality.

## Data locality restoration

If the blocks of the file are not located as what you want, restore or change the locality of the file.

The IBM Spectrum Scale FPO provides interface for you to control all first replica of the blocks, all second replicas of the blocks, and all third replicas of the blocks in specific nodes. For example, you can have the first replica of all blocks located in a specific node so that the applications running over the node can read all data from local disks.

**Note:** The IBM Spectrum Scale FPO does not support the control of the location of only one or part of blocks. For example, you cannot control the location of block 1 or block 2 without changing the location of block 3.

**Restoring the locality for files without WADFG:**

This topic lists the steps to control the first replica of all blocks.

1. Check whether the file is configured with WADFG.

```
mmlsattr -d -L /sncfs/test
file name:            /sncfs/test
metadata replication: 3 max 3
data replication:     3 max 3
immutable:            no
appendOnly:           no
flags:
storage pool name:    datapool
fileset name:         root
snapshot name:
Write Affinity Depth Failure Group(FG) Map for copy:1 1,0,1
Write Affinity Depth Failure Group(FG) Map for copy:2 2,0,1
Write Affinity Depth Failure Group(FG) Map for copy:3 3,0,1
creation time:        Thu Mar 24 16:15:01 2016
Misc attributes:      ARCHIVE
Encrypted:            no
gpfs.WADFG:           0x312C302C313B322C302C313B332C302C31
```

   If you see gpfs.WADFG (as per the preceding example) from the output of **mmlsattr**, the file is configured with WADFG, and, in this gpfs.WADFG case, follow the instructions in "Restoring the locality for files with WADFG" on page 472. If you do not see the gpfs.WADFG text, go to the step2.
2. Select the node to store all the blocks from the first replica of the data. One disk from this node is used to store the first replica of the file, assuming that this node has at least one local disk that serves the GPFS filesystem.
3. If you are using Spectrum Scale 4.1.1.0 or later:
   a. ssh to the target node selected in step 2
   b. run **mmrestripefile –l filename** for each filename to set the data locality for.

   If you are using Spectrum Scale 4.1.1.0 or earlier
   a. ssh to the target node selected in step 2
   b. **mmchdisk <fs-name> suspend -d "any-one-data-disk"**
   c. **mmchdisk <fs-name> resume -d "any-one-data-disk"**

d. **mmrestripefile -b filename**

4. Check the data locality by running:

   ```
   /usr/lpp/mmfs/samples/fpo/tsGetDataBlk /sncfs/test -r 3
   ```

   The first replica of all blocks is located in the target node.

**Restoring the locality for files with WADFG:**

If you want to control the location of the first replica, second replica, and the third replica, set the WADFG attributes of the files via `mmchattr`. If you are using IBM Spectrum Scale 4.1.1.0 or earlier, perform these steps to restore the data locality.

1. Decide the location for data replica.
2. Run **mmchattr --write-affinity-failure-group** to set/update the new WADFG of the file Step.
3. If you are using IBM Spectrum Scale 4.1.1.0 or later, skip this step. Run **mmrestripefile –l filename** or **mmrestripefile –b filename**.

   In IBM Spectrum Scale 4.1.1.0 and later, the default option for **mmchattr** is `-I yes`, and the restripe function of'**mmrestripefile -l** is performed when **mmchattr** is run.
4. Check the data locality.

# Disk Replacement

This topic describes how to replace a disk.

In a production cluster, you can replace physically broken disks with new disks or replace the failed disks with new disks.

- If you have non functional disks from two failure groups for replica 3, restripe the file system to protect the data to avoid data loss from a third non functional disk from the third failure group.
- Replacing the disks is time-consuming because the whole inode space must be scanned and the IO traffic in the cluster is triggered. Therefore, schedule the disk replacement when the cluster is not busy.

The **mmrpldisk** command can be used to replace one disk in file system with a new disk and it can handle one disk in one invokation. If you want to replace only one disk, see **mmrpldisk** command.

**Note:** In FPO, sometimes **mmrpldisk** command does not migrate all data from the to-be-replaced disk to the newly added disk. This bug impacts IBM Spectrum Scale Release 3.5 and later. See the following example:

```
[root@c8f2n03 ~]# mmlsdisk sncfs –L
disk         driver   sector     failure holds    holds                                     storage
name         type       size       group metadata data  status        availability disk id pool         remarks
------------ -------- ------ ----------- -------- ----- ------------- ------------ ------- ------------ ---------
n03_0        nsd         512           1 Yes      Yes   ready         up                 1 system
n03_1        nsd         512           1 Yes      Yes   ready         up                 2 system        desc
n04_0        nsd         512       2,0,0 Yes      Yes   ready         up                 3 system        desc
n04_1        nsd         512       2,0,0 Yes      Yes   ready         up                 4 system
n05_1        nsd         512       4,0,0 No       Yes   ready         up                 5 system        desc
Number of quorum disks: 3
Read quorum value:      2
Write quorum value:     2

[root@c8f2n03 ~]# /usr/lpp/mmfs/samples/fpo/tsGetDataBlk /sncfs/log -s 0
File length: 1073741824, Block Size: 1048576
Parameters: startoffset:0, skipfactor: META_BLOCK, length: 1073741824, replicas 0
numReplicasReturned: 2, numBlksReturned: 8, META_BLOCK size: 134217728
Block 0 (offset 0) is located at disks:  2   5
Block 1 (offset 134217728) is located at disks:  2   3
Block 2 (offset 268435456) is located at disks:  2   5
Block 3 (offset 402653184) is located at disks:  2   3
Block 4 (offset 536870912) is located at disks:  2   5
Block 5 (offset 671088640) is located at disks:  2   3
```

```
Block 6 (offset 805306368) is located at disks:  2   5
Block 7 (offset 939524096) is located at disks:  2   3

[root@c8f2n03 ~]# mmrpldisk sncfs n03_1 n03_4
[root@c8f2n03 ~]# mmlsdisk sncfs –L
disk         driver   sector     failure holds    holds                                     storage
name         type       size       group metadata data  status        availability disk id pool         remarks
------------ -------- ------ ----------- -------- ----- ------------- ------------ ------- ------------ ---------
n03_0        nsd         512           1 Yes      Yes   ready         up                 1 system       desc
n04_0        nsd         512       2,0,0 Yes      Yes   ready         up                 3 system       desc
n04_1        nsd         512       2,0,0 Yes      Yes   ready         up                 4 system
n05_1        nsd         512       4,0,0 No       Yes   ready         up                 5 system       desc
n03_4        nsd         512           1 Yes      Yes   ready         up                 6 system
Number of quorum disks: 3
Read quorum value:      2
Write quorum value:     2
[root@c8f2n03 ~]# /usr/lpp/mmfs/samples/fpo/tsGetDataBlk /sncfs/log -s 0
File length: 1073741824, Block Size: 1048576
Parameters: startoffset:0, skipfactor: META_BLOCK, length: 1073741824, replicas 0
numReplicasReturned: 2, numBlksReturned: 8, META_BLOCK size: 134217728
Block 0 (offset 0) is located at disks:  6   5
Block 1 (offset 134217728) is located at disks:  1   3
Block 2 (offset 268435456) is located at disks:  1   5
Block 3 (offset 402653184) is located at disks:  1   3
Block 4 (offset 536870912) is located at disks:  6   5
Block 5 (offset 671088640) is located at disks:  6   3
Block 6 (offset 805306368) is located at disks:  1   5
Block 7 (offset 939524096) is located at disks:  6   3
After replacing n03_1 with n03_4, part of data located in n03_1 are migrated into n03_4
and others are migrated into n03_0. Therefore, mmrpldisk doesn't mean copy data from the
to-be-replaced disks into new added disks. mmrpldisk might break the data locality and
you need to see the Section 9 to restore data locality if needed.
```

If you want to replace more than one disk, run the **mmrpldisk** command multiple times. The PIT job is triggered to scan the whole inode space to migrate the data to disks that are going to be replaced. The IO traffic is triggered and is time-consuming if you have to run the **mmrpldisk** command multiple times. To speed the replacement process, see the following sub sections to replace more than one disk in the file system.

## Replace more than one active disks

This topic describes how to replace more than one active disk.

If you want to replace more than one disk used in file system, and if you have a lot of files or data in the file system, it will take long time to do this if you are using the **mmrpldisk** command for each disk.

If you have additional idle disk slots, you can plug new disks into these idle slots and run **mmcrnsd** to create new NSD disks against the disks that are to be added, run **mmadddisk** (without the option -r) to add the new disks into the file system and then **mmdeldisk** the disks that are to be replaced by using **mmdeldisk**.

**Note:** If you place new disks in the same failure group of the disks that are to be replaced, the above operations will maintain the data locality for the data from disks that are to be replaced. IBM Spectrum Scale keeps the data in the original failure group.

If you do not have additional idle disk slots, run the **mmdeldisk** command on the disks that are to be replaced, run **mmcrnsd** to create the NSD disks and run **mmadddisk** to add the NSD disks to the file system. You might have to run **mmrestripefs -b** to balance the file system but this breaks the data locality.

## Replace more than one broken disks

If you want to replace more than one disk that are physically broken, you cannot read any data from these disks or write any data into these disks,

if the broken disks have been restriped they become emptied or non functional. Run the **mmdeldisk** command directly. Pull out the broken disks, pull in the new disks, run **mmcrnsd**, and then run **mmadddisk** to add them into the file system.

If the broken disks have not been restriped (then, it might be ready/down or ready/up), then take the following steps:

1. Disable auto recovery temporarily (refer the section 2.1, step 2)
2. Pull out the broken disks directly. You could run **mmlsnd -X** to check what these pulled-out disks will be like: `node7_sdn C0A80A0756FBAA89 - - gpfstest7.cn.ibm.com (not found) server node`
3. Pull in the new disks.
4. **mmcrnsd** for the new disks (take new NSD name)
5. **mmadddisk <fs-name> -F <new-nsd-file from step4>**
6. To delete the broken disks from the file system, see *Disk media failure* in *IBM Spectrum Scale: Problem Determination Guide*.

---

# Auto Recovery for Disk Failure

The FPO-enabled/disabled storage pool over internal disks are subject to frequent node and disk failures because of the commodity hardware used in IBM Spectrum Scale clusters.

IBM Spectrum Scale auto recovery feature is designed to handle random but routine node and disk failures without requiring manual intervention. However, auto recovery cannot cover all catastrophic outages involving large number of nodes and disks at once. Administrator assessment of the situation and judgment is required to determine the cluster recovery action.

# Failure and recovery

There are two main failures to consider for FPO environments.

1. Node failure and outages: These outages include reboot, kernel crash and hang and can last long. When a node is inaccessible, all the associated disks also become inaccessible.
2. Disk failures: These failures include disk failures, hard IO errors and are generally triggered by hardware failures and affect specific disks.

IBM Spectrum Scale recovery actions are enabled by setting the `restripeOnDiskFailure` configuration option to yes. When this option is enabled, auto recovery leverages the IBM Spectrum Scale event callback mechanism to trigger necessary actions to perform recovery actions. Specifically, the following system callbacks are installed when `restripeOnDiskFailure=yes`.

- **event = diskFailure action: /usr/lpp/mmfs/bin/mmcommon recoverFailedDisk %fsName %diskName**
- **event = nodeJoin action: /usr/lpp/mmfs/bin/mmcommon restartDownDisks %myNode %clusterManager %eventNode**
- **event = nodeLeave action: /usr/lpp/mmfs/bin/mmcommon stopFailedDisk %myNode %clusterManager %eventNode**

## diskFailure Event

This event is triggered when a disk I/O operation fails. Upon I/O failure, IBM Spectrum Scale marks the disk from read/up to ready/down. This I/O failure can also be caused by a node, because all disks connected by the node become unavailable, or a disk failure.

The disk state is ready/down.

### Recovery process

1. Perform simple checks, such as fpo pool and replication >1.

2. Check the **maxDownDisksForRecovery** (default16), **maxFailedNodesForRecovery** (default 3). Abort if the limit is exceeded. Note that these limits can be changed by using the configuration parameters.
3. If the number of failed FGs is less than 2, wait until **dataDiskWaitTimeForRecovery** (default 3600/2400) expires, otherwise wait for **minDiskWaitTimeForRecovery** (default 1800 sec) to expedite recovery due to increased risk.
4. If the available FGs for metadata is less than three, no action is taken because recovery cannot be performed due to the metadata outage.
5. After the recovery wait period has passed, recheck the node and disk availability status to ensure that recovery actions are taken.
6. Suspend all the failed and unavailable disks by running the **tschdisk suspend** command.
7. Restripe the data. If a previous restripe process is running, stop it and start a new process.
8. At successful completion, disks will be in suspended/down or suspended/up if the node is recovered during the restripe.

## nodeJoin Event

This event is triggered when a node joins the cluster after a node reboot or rejoined after losing membership to the cluster or getting started after an extended outage. Scope of the recovery is all file systems to which the node disks might belong to. In most case, the disk state can be ready/up if no I/O operation has been performed or ready/down. However, based on the prior events, the state could vary to suspended/down or unrecovered/recovering.

### Recovery process

1. Perform simple checks on the disks assigned to the file systems.
2. Check if a **tschdisk** start is already running from a prior event. Kill the process to include disks from the current nodes.
3. Start all disks on all nodes by running: **tschdisk start -a** to optimize recovery time. This command requires all nodes in the cluster to be functioning in order to access all the disks in the file system.
4. Start All down disks on all Active nodes by running: **tschdisk start -F<file containing disk list>**.
5. After successful completion, all disks must be in the ready/up or the suspended/up state if the node is being recovered from a long outage.

If a new diskFailure event is triggered while **tschdisk** start is in progress, the disks will not be restored to the Up state until the node joins the cluster and triggers a nodeJoin event.

## nodeLeave Event

This event is triggered when a node leaves the cluster, is expelled, or shut down.

The processing of this event is similar to the diskFailure event, except that disks may not already be marked as Down when this event is received. Note that a diskFailure event can still be generated based on an I/O activity in the cluster. If it is generated, no action will be taken by the diskFailure event handler if the owning node is also down, thereby allowing the nodeLeave event to control the recovery. In most cases, the disk state could be ready/up if no I/O operation has been performed or ready/down. However, based on prior events the state could be suspended/down or unrecovered/recovering.

### Recovery process

1. Wait for the specified duration to give the failed nodes a chance to recover.
2. Check the Down nodes count, Down disks count and available data and metadata FG count to check against the maximum limit.
3. Build a list of disk to act upon, ignoring suspended, empty, to be emptied.
4. Run **tsrestripefs** to restore replica count to the stated values.

5. After successful completion, disks can be in the suspended/down state or no action may be taken if the nodeJoin event is triggered within the **recoveryWaitPeriod**.

## Important Notes to the administrator

Following are some important notes for the administrator:

- Once disks are suspended by the auto recovery, these disks must be manually resumed by the administrator so that disks can be used by the file system. When the owning node is determined to be the healthy again, **mmchdisk resume** command must be run.
- If extended outages (days and weeks) are expected, it is recommended to remove that node and all associated disks from the cluster to avoid this outage from affecting subsequent recovery actions.
- If the failed disk is meta disk, during auto recovery, it will try to **mmchdisk <file-system>** suspend the failed disk. If the remaining failure groups of meta or data disks is less than the value of -r/-m, this will make **mmchdisk <file-system>** suspend/fail, and therefore auto recovery will not take further actions.

---

## Restrictions

An FPO environment includes restrictions.

The following restrictions apply:

- Storage pool properties can be set only when the pool is created and cannot be changed later.
- All disks in an FPO pool must be assigned an explicit failure group.
- All disks in an FPO pool must have exactly one NSD server associated with them.
- All disks in an FPO pool that share an NSD server must belong to the same failure group.
- When replacing a disk in an FPO pool, the old and new disks must have the same NSD server.
- Disks must be removed from the file system before NSD servers can be changed.
- FPO is not supported on Windows.

There might be additional limitations and restrictions. For the latest support information, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

# Chapter 32. Hadoop support for IBM Spectrum Scale

The Apache Hadoop framework features open source software that enables distributed processing of large data sets across clusters of commodity servers. It is designed to scale up from a single server to thousands of machines, with a high degree of fault tolerance.

The Apache Hadoop framework allows distributed processing of large data sets across clusters of computers that use simple programming models. The Apache Hadoop framework processes data-intensive computational tasks, which include data amounts that can range from hundreds of terabytes (TBs) to tens of petabytes (PBs). This computation mode differs from the computation mode that is used in traditional high-performance computing (HPC) environments.

For example, Hadoop consists of many open source modules. One of the primary open source modules is the Hadoop Distributed File System (HDFS), which is a distributed file system that runs on commodity hardware. HDFS lacks enterprise class capabilities necessary for reliability, data management, and data governance. IBM Spectrum Scale, which is a scale-out distributed file system, offers an enterprise-class alternative to HDFS.

### Hadoop collaboration with IBM Spectrum Scale

IBM Spectrum Scale provides integration with Hadoop applications that use the Hadoop connector (so you can use IBM Spectrum Scale enterprise-level functions on Hadoop):

- POSIX-compliant APIs or the command line
- FIPS and NIST compliant data encryption
- Disaster Recovery
- Simplified data workflow across applications
- Snapshot support for point-in-time data captures
- Simplified capacity management by using IBM Spectrum Scale (for all storage needs)
- Policy-based information lifecycle management capabilities to manage PBs of data
- Infrastructure to manage multi-tenant Hadoop clusters based on service-level agreements (SLAs)
- Simplified administration and automated recovery
- Multiple clusters

## Hadoop connector

This section further describes how the Hadoop connector enables Hadoop for IBM Spectrum Scale.

### Hadoop connector support for IBM Spectrum Scale

The IBM Spectrum Scale Hadoop connector, which must be installed on each Hadoop node, implements Hadoop file system APIs and the `FileContext` class so it can access the IBM Spectrum Scale.

The IBM Spectrum Scale Hadoop connector is composed of these parts:

- `hadoop-gpfs-*.jar`
- `libgpfshadoop.so`
- `gpfs-connector-daemon`

Install the connector on each node in the Hadoop cluster.

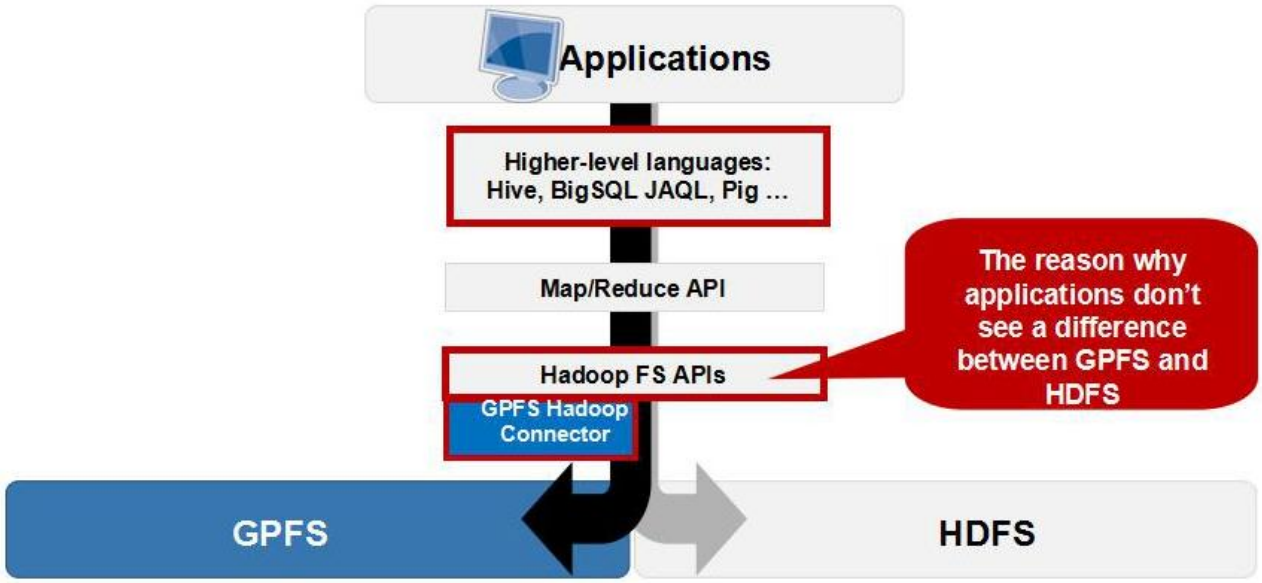



*Figure 15. GPFS Hadoop connector overview*

**Note:** Specify the **/usr/lpp/mmfs/bin/mmhadoopctl** command to deploy the connector on Hadoop and to control the connector daemon.

## Hadoop support for the IBM Spectrum Scale storage mode

IBM Spectrum Scale has two storage modes that Hadoop can access:

- Centralized Storage Mode
- Local Storage Mode, or File Placement Optimizer (FPO) Mode

IBM Spectrum Scale allows Hadoop applications to access centralized storage data like the SAN Volume Controller, which means that storage is attached to a dedicated storage server (or accessed directly by SAN Volume Controller). All Hadoop replica nodes can access the storage as a IBM Spectrum Scale client. You can share a cluster between Hadoop and any other application.

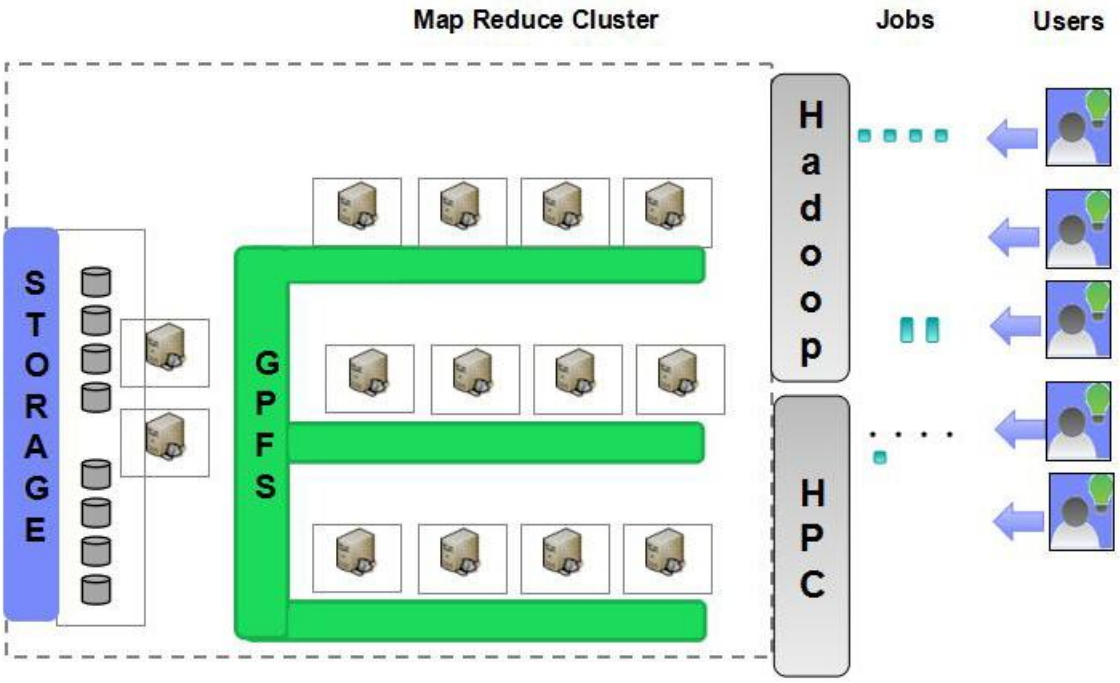



*Figure 16. Hadoop on centralized storage*

IBM Spectrum Scale also provides a local storage mode, which is FPO (for Hadoop). FPO is an implementation of shared-nothing architecture that enables each node to operate independently, which reduces the impact of failure events that occur across multiple nodes.

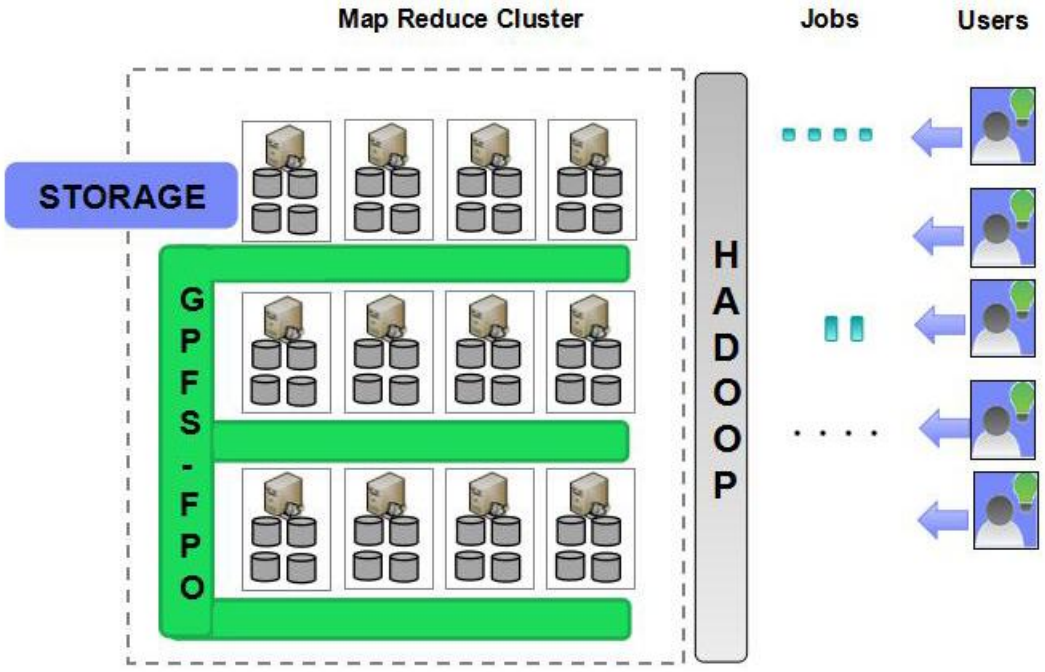



*Figure 17. Hadoop on Local storage (FPO)*

## IBM Spectrum Scale cluster planning for Hadoop applications

When you are creating a IBM Spectrum Scale file system for a Hadoop application:

- IBM Spectrum Scale allows Hadoop to run on a file system with multiple storage pools. For purposes of storage pool planning, these storage pools can be either generic or FPO (with `allowWriteAffinity=yes`). With central storage, IBM Spectrum Scale can run Hadoop on a typical file system configuration that has a mixed storage pool with both metadata and data.
- You must consider replica and block size planning:

*Table 41. Replica and block size planning*

| Mode | Data type | Block size | Replica |
|---|---|---|---|
| FPO mode | Metadata | A small size is suggested. For example, 256 KB. | 2 or 3<br><br>Fewer replicas apply to disks with hardware protection such as RAID. |
| | Data | A large size is suggested. A typical value is 2 MB. | 3<br><br>Replica 2 is only considered with extra disk protection such as RAID. |
| Central Storage mode | Metadata | It depends on the application I/O pattern and whether the application is using RAID or not. | 1 or 2 |
| | Data | It depends on the application I/O pattern and whether the application is using RAID or not. | 1 or 2 |

- You can use any advanced features in IBM Spectrum Scale, such as Local Read-Only Cache (LROC), to improve performance.
- Hadoop can run on a remote file system that is mounted from another cluster.

## Hadoop cluster planning

In an IBM Spectrum Scale shared storage environment, do not use an NSD server as a computing node because NSD servers generally maintain a heavy I/O workload.

**Remember:** In a Scale Central Storage (SCS) environment, any node can be used as a Hadoop replica node.

## IBM Spectrum Scale functions corresponding to HDFS

To enable HDFS-4685 access control lists (ACLs):

1. Enable GPFS Posix ACL support:

```
# mmchfs bigfs -k posix
# mmlsfs bigfs -k

flag                value                    description
----                -----                    -----------
-k                  posix                    ACL semantics in effect
```

2. Install the dependent packages:
   - `acl`
   - `libacl`
   - `libacl-devel` (required by Hadoop 2.4 connector only)

In this example, the IBM Spectrum Scale file system name is `bigfs`.

To enable the HDFS-2006 ability to store extended attributes per file, install the dependent `libattr` package.

To enable the remote file system in the Hadoop connector, add the following property:

```
<property>
<name>gpfs.remote.cluster.enabled</name>
<value>true</value>
</property>
```

## Installing the IBM Spectrum Scale Hadoop connector

IBM Spectrum Scale Hadoop connector 2.7 is installed under the `/usr/lpp/mmfs/hadoop` directory.

See the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html) to get the list of Hadoop versions supported by IBM Spectrum Scale Hadoop connector. It is highly recommended that you use the latest version of Hadoop connector.

If you are running old Hadoop, you need to install old IBM Spectrum Scale Hadoop connector version.

Different Hadoop distributions and versions might require different methods to deploy and configure the IBM Spectrum Scale Hadoop connector. If you are using IBM BigInsights 3.0.x or any earlier release, IBM Spectrum Scale Hadoop connector can be installed and configured by IBM BigInsights automatically. If you are using IBM Platform Symphony with IBM Spectrum Scale, follow the procedure described in the IBM Platform Integration Guide for MapReduce Applications to install IBM Platform Symphony, and then configure IBM Spectrum Scale Hadoop connector by following the guide shipped with IBM Platform Symphony. If you are using IOP 4.x, you can use Ambari to install and configure IBM Spectrum Scale cluster and connector automatically. For more information, see the "Deploy IBM Spectrum Scale using Ambari" on page 491 topic. If you are using Hadoop distribution from Apache or other vendors, check your Hadoop distribution document to check the prerequisites, and then deploy the IBM Spectrum Scale and connector.

To install the IBM Spectrum Scale Hadoop 2.7 connector, download it from the IBM developerWorks® Wiki.

If you are using Hadoop-2.7 version, do the following steps to install and configuring the IBM Spectrum Scale Hadoop connector.

1. Run the following command to install IBM Spectrum Scale Hadoop connector:
   ```
   mmhadoopctl connector attach --distribution <BigInsights or Apache> -N all
   ```
2. Start IBM Spectrum Scale Hadoop connector by running the following command:
   ```
   mmhadoopctl connector start -N all
   ```
3. Check the IBM Spectrum Scale Hadoop connector health by running the following command:
   ```
   mmhadoopctl connector getstate -N all
   ```

**Note:** The `–N` option is only supported after IBM Spectrum Scale 4.1.1 PTF1 release and later.

For additional applicable configuration change to Hadoop configuration files, see the topic "Modifying the Hadoop configuration to use IBM Spectrum Scale."

## Modifying the Hadoop configuration to use IBM Spectrum Scale

This section describes configuration updates to the Hadoop MapReduce applications. Regardless of the Hadoop distribution being used, you should see the settings below and make the required changes before running MapReduce applications.

The following configuration changes are for Hadoop 2.7 when taking BigInsights 4.0/4.1. When taking other Hadoop distributions, you need to check the configuration against your Hadoop distributions and update the configuration by using the interface provided by the vendor.

Hadoop component storage falls into two main categories:

- **Shared distributed storage**: Referred to as the Distributed File System in Hadoop documentation and configuration files, these paths map to the IBM Spectrum Scale file system. By default, Hadoop configuration files are defined to use HDFS and need to be updated as described below.
- **Local storage**: This term refers to the storage managed as the file system locally on the node. Paths are specified for local storage map to a local file system on the node such as `ext3`, `ext4`. Local storage is used to keep log files generated by Hadoop components and other temporary files that do not need to be distributed. Using the local storage for temporary files can provide better performance, as it avoids replication overhead for metadata and data.

**Note:** If you are using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration file directly. IBM BigInsights uses a database to manage Hadoop configurations. Hadoop configuration files would be refreshed when IBM BigInsights services are restarted. You must stop services before applying the below changes and then start service for the changes to take effect.

The `core-site.xml` file contains the site-specific configuration for a given Hadoop installation.

You must stop HDFS services before applying the below changes and start HDFS service for the changes to take effect, and then stop HDFS server again. For HDFS service, check the Summary section to ensure HDFS related services are down. Ignore any failure message here since this step needs to only save changes into IBM BigInsights repository and configuration files.

*Table 42. Modifications to Hadoop configuration—core-site.xml*

| Name | Value | Notes |
|---|---|---|
| `fs.hdfs.impl` | See notes | This configuration specifies the IBM Spectrum Scale connector class name.<br><br>Add the configuration through web GUI.<br><br>Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to `org.apache.hadoop.fs.gpfs.GeneralParallelFileSystem` value . |
| `fs.gpfs.impl` | See notes | This configuration specifies the IBM Spectrum Scale connector class name.<br><br>Add the configuration through web GUI.<br><br>Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to `org.apache.hadoop.fs.gpfs.GeneralParallelFileSystem` value . |
| `gpfs.mount.dir` | absolute path to IBM Spectrum Scale file system mount directory | This configuration specifies the mount point of the IBM Spectrum Scale file system on all nodes.<br><br>For example, under `/gpfs/bigfs`. Add the configuration through web GUI. Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to the absolute path to local file system mount directory. |

*Table 42. Modifications to Hadoop configuration—core-site.xml (continued)*

| Name | Value | Notes |
|---|---|---|
| `hadoop.tmp.dir` | absolute path to local file system mount directory | This configuration specifies a path on the local file system. This is used as a base directory for local Hadoop files, task output and error files, and task-specific Java™ class. This path must exist on each node.<br><br>For example, under `/hadoop/local/sd1/tmp`. Add the configuration through web GUI. Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to the absolute path to local file system mount directory. |
| `fs.AbstractFileSystem.hdfs.impl` | See notes | Add the configuration through web GUI. Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to the `org.apache.hadoop.fs.gpfs.GeneralParallelFs` value. |
| `fs.AbstractFileSystem.gpfs.impl` | See notes | Add the configuration through web GUI. Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to the `org.apache.hadoop.fs.gpfs.GeneralParallelFs` value. |
| `gpfs.supergroup` | hadoop | Configure the groups that are privileged on the file system. Multiple groups are separated by comma. There are two kinds of user groups in Hadoop, one is super user group while the other is normal user. The super user group must have super permission to access distributed file system. All request from super user will redirect to IBM Spectrum Scale Hadoop connector and executed as root user. All the other user's request are handled locally by JVM.<br><br>Add the configuration through web GUI. Choose **HDFS service** > **Configs**, in the **Custom core-site** section, add this new configuration and set it to the specific value.<br>**Note:** The group **`hadoop`** is the group name of all users from a different service, for example, hbase, hive, and yarn. If the customer takes a non-default group name for these different service users, then the **`gpfs.supergroup`** must be the group name used by the customer. |
| `gpfs.remote.cluster.enabled` | true | IBM Spectrum Scale support multiple clusters, so a file system might mount from a remote cluster. You can run Hadoop on this sort of cluster by enabling the configuration. As the local cluster cannot make use of data locality in remote filesystem, connector reports an arbitrary hostname for data block location. The hostname for this usage is **`gpfsNSDserver`** by default. Do not use this reserved name as any host name in cluster as then Hadoop locates more tasks on the **`gpfsNSDserver`** node, which makes the workload unbalanced.<br>**Note:** This configuration is not required if the file system is not mounted remotely. |

If you are using IBM Platform Symphony or another Apache Hadoop distribution, the file must be located under `<Hadoop install path>/conf/`. The file must be updated on each node.

The `mapred-site.xml` file contains configuration information that overrides the default values for **`MapReduce`** parameters.

If you are using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration file directly. You must stop **MapReduce2** services before applying following changes and start **MapReduce2** service for the changes to take effect. If you are using IBM Platform Symphony or another Apache Hadoop distribution, the file must be located under `<Hadoop install path>/conf/`. The file must be updated on each node.

*Table 43. Modifications to Hadoop configuration—mapred-site.xml*

| Name | Value | Notes |
|---|---|---|
| `mapred.cluster.local.dir` | See notes | This is the local directory where MapReduce stores intermediate data files. It migth be a comma-separated list of directories on different devices in to spread disk I/O.<br><br>Directories that do not exist are ignored. If multiple disk partitions were created during preparation, use those partitions by specifying a path on each local file system on the second disk partition. This configuration controls only temporary file from MapReduce V1 jobs. YARN job temporary files location are controlled by configurations mentioned in Table 44 on page 485.<br><br>For example, In the `/hadoop/local/sd1/local,/hadoop/local/sd2/local,/hadoop/local/sd3/local`, add the configuration through web GUI.<br><br>Choose **MapReduce2 service > Configs** in **Custom mapred-site** section.<br><br>Add **mapred.cluster.local.dir** configuration and set it to the absolute path in local file system mount directory where you want to store MapReduce intermediate data files. It is not supported to save MapReduce intermediate data into the IBM Spectrum Scale file system if using IBM BigInsights.<br>**Note:** This variable is only effective for Hadoop MR V1 jobs. For MR2, see the Yarn's configuration. |
| `mapreduce.map.java.opts`, `mapreduce.reduce.java.opts` and `yarn.app.mapreduce.am.command-opts` | `-Xmx1229m` (See notes) | These values specify the heap memory used by the Java Virtual Machine running MapReduce task. It must be set based on the amount of memory on each computing node after 25 percent is set aside for the IBM Spectrum Scale page pool. It also helps determine the number of tasks that can be run in parallel on each node.<br><br>For example, on a node with 48 GB memory, 1,000m per task would approximately allow 18 map tasks and 3 reduce tasks. Update the configuration through web GUI.<br><br>Choose **MapReduce2 service > Configs**, in the **Advanced mapred-site** section, Update `mapreduce.map.java.opts/ mapreduce.reduce.java.opts/ yarn.app.mapreduce.am.command-opts` if these values do not fit your system configuration. |
| `mapreduce.application.classpath` | See notes | Update the configuration through web GUI.<br><br>Choose **MapReduce2 service > Configs**, in the **Advanced mapred-site** section,, add the string `/usr/iop/current/hadoop-client/hadoop-gpfs.jar` into the beginning of existing value. |
| `mapreduce.application.framework.path` | Keep the default value. See notes. | If your MapReduce job does not find frame work package, ensure IBM Spectrum Scale file system is mounted, HDFS service is down, MapReduce2 and YARN service are up, and run the following command to resolve the problem:<br>`hadoop fs -put /usr/iop/$IOP_VERSION/hadoop/mapreduce.tar.gz /iop/apps/$IOP_VERSION/mapreduce/mapreduce.tar.gz` |

For YARN, site-specific customization is done by specifying overriding parameters in the `Yarn-site.xml` file.

If using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration file directly. You must stop **Yarn** services before applying following changes and start **Yarn** service for the changes to take effect.

*Table 44. Modifications to Hadoop configuration—Yarn-site.xml*

| Name | Value | Notes |
|---|---|---|
| yarn.nodemanager.local-dirs | See notes | This is the local directory where YARN stores intermediate data files. It migth be a comma-separated list of directories on different devices to spread disk I/O. Directories that do not exist are ignored. If multiple disk partitions were created during preparation, use those partitions by specifying a path on each local file system on the second disk partition. This configuration only controls temp file from YARN jobs.<br><br>For example: in the `/hadoop/local/sd1/local,/hadoop/local/sd2/local,/hadoop/local/sd3/local` directory. Add the configuration through web GUI. Choose **MapReduce2** service **> Configs**, in the **Custom yarn-site** section, add the **yarn.nodemanager.local-dirs** configuration and set it to the absolute path in local file system mount directory where you want to store YARN intermediate data files. You cannot save YARN's intermediate data into the IBM Spectrum Scale file system if using IBM BigInsights.<br><br>If you want to put the data into the shared storage, you need to take the following steps:<br>1. Create a fileset named `local` by running the following command:<br>`mmcrfileset <gpfs-fs-name> local`<br>In the examples the name of the directory is local.<br>2. Link the local fileset to a directory in the GPFS filesystem by running the following command:<br>`mmlinkfileset <gpfs-fs-name> local -J <gpfs-mount-point>/local`<br>3. Create a directory for each host in the Hadoop cluster under the junction path by running the following command:<br>`mkdir <gpfs-mount-point>/local/<hostnameX>`<br>The following examples show how to create directories:<br>`mkdir /gpfs/local/host1, mkdir /gpfs/local/host2, mkdir/gpfs/local/host3`<br>4. On each host, create a symbolic link between one of the directories to a directory on the local filesystem.<br>5. Ensure that the directory does not exist on the local filesystem by running the following command:<br>`ln -s /<gpfs-mount-point>/local/<hostnameX> /hadoop/local/local_dir`<br>6. Create a policy with the local fileset with one replica for reduced impact on performance.<br>`/* Local policy with one replica */`<br>`RULE 'local' SET POOL 'datapool' REPLICATE (1,1) FOR FILESET (local)`<br>`/* Default placement policy rule */`<br>`RULE 'default' SET POOL 'datapool'`<br>7. Test the policy for errors by running the following command:<br>`mmchpolicy <gpfs-fs-name> <policy-name> -I test`<br>8. If there are no errors, apply the policy by running the following command:<br>`mmchpolicy <gpfs-fs-name> <policy-name> -I yes`<br>9. Configure `yarn.nodemanager.local-dir` as `/hadoop/local/local_dir` in Ambari GUI. |

*Table 44. Modifications to Hadoop configuration—Yarn-site.xml (continued)*

| Name | Value | Notes |
|---|---|---|
| yarn.nodemanager.log-dirs | See notes | This is the local directory where YARN stores log files.<br><br>Either this can be a local directory where YARN stores log files in each node or it can be a directory from IBM Spectrum Scale file system as it is then easy to check YARN log from all nodes in one place.<br><br>For example: in the `/hadoop/local/sd1/local,/hadoop/local/sd2/local,/hadoop/local/sd3/local` directory. Add the configuration through web GUI. Choose **MapReduce2** service **> Configs**, in the **Custom yarn-site** section, add the **yarn.nodemanager.logs-dirs** configuration and set it to the absolute path in local file system mount directory where you want to store YARN log files.<br><br>If you want to put the data into the shared storage, you need to take the following steps:<br>1. Create a fileset named `local` by running the following command:<br>`mmcrfileset <gpfs-fs-name> local`<br>In the examples the name of the directory is local.<br>2. Link the local fileset to a directory in the GPFS filesystem by running the following command:<br>`mmlinkfileset <gpfs-fs-name> local -J <gpfs-mount-point>/local`<br>3. Create a directory for each host in the Hadoop cluster under the junction path by running the following command:<br>`mkdir <gpfs-mount-point>/local/<hostnameX>`<br>The following examples show how to create directories:<br>`mkdir /gpfs/local/host1, mkdir /gpfs/local/host2, mkdir/gpfs/local/host3`<br>4. On each host, create a symbolic link between one of the directories in the Hadoop cluster to a directory on the local filesystem.<br>5. Ensure that the directory does not exist on the local filesystem by running the following command:<br>`ln -s /<gpfs-mount-point>/local/<hostnameX> /hadoop/local/local_dir`<br>6. Create a policy with the local fileset with one replica for reduced impact on performance.<br>`/* Local policy with one replica */`<br>`RULE 'local' SET POOL 'datapool' REPLICATE (1,1) FOR FILESET (local)`<br>`/* Default placement policy rule */`<br>`RULE 'default' SET POOL 'datapool'`<br>7. Test the policy for errors by running the following command:<br>`mmchpolicy <gpfs-fs-name> <policy-name> -I test`<br>8. If there are no errors, apply the policy by running the following command:<br>`mmchpolicy <gpfs-fs-name> <policy-name> -I yes`<br>9. Configure `yarn.nodemanager.log-dirs` as `/hadoop/local/local_dir` in Ambari GUI. |

For Hive, site-specific customization is done by specifying overriding parameters in the `hive-site.xml` file.

If using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration files directly.

You must stop the **Hive** services before applying the following changes and start the **Hive** service for the changes to take effect. If using IBM Platform Symphony or another Apache Hadoop distribution, the file must be located under the `<hive_home>/conf/` folder. The file must be updated on each node.

*Table 45. Modifications to Hadoop configuration—hive-site.xml*

| Name | Value | Notes |
|---|---|---|
| `hive.exec.scratchdir` | relative path in GPFS file system | This value specifies a path in the IBM Spectrum Scale file system to be used for temporary data. This can be defined on a fileset that has the replication factor set to 1 through policy rules. The path specified is relative to the IBM Spectrum Scale file system mount point.<br><br>For example, in the `/hive-scratch` directory, update the configuration through web GUI.<br><br>Choose **Hive** service **> Configs**, in the **Advanced hive-site** section, update this configuration to the absolute link path of the IBM Spectrum Scale fileset if you decide to use IBM Spectrum Scale to save Hive temporary data.<br><br>If you are using other user, such as hive, instead of the root user to start the **Hive**service, you need to change the owner and other related permission of this directory to specify the user. For example,<br>`chown hive:hadoop /hive-scratch`<br>`chmod 733 /hive-scratch` |

For Ooize, site-specific customization is done by specifying overriding parameters in the `Ooize-site.xml` file.

If using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration file directly.

You must stop **Ooize** services before applying the following changes, and then start **Ooize** service for the changes to take effect.

*Table 46. Modifications to Hadoop configuration—Ooize-site.xml*

| Name | Value | Notes |
|---|---|---|
| `oozie.service.HadoopAccessorService.supported.filesystems` | * | This value specifies supported file systems for Ooize services.<br><br>Update the configuration through web GUI.<br><br>Choose **Ooize** service **> Configs**, in **Custom ooize-site** section, update this configuration to value "*". |

For HBase, site-specific customization is done by specifying overriding parameters in the `hbase-site.xml` file.

If using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying configuration file directly.

You must stop **HBase** services before applying the following changes and start **HBase** service for the changes to take effect. If using IBM Platform Symphony or another Apache Hadoop distribution, the file must be located under `<hbase home>/conf/` directory. The file must be updated on each node.

*Table 47. Modifications to Hadoop configuration—hbase-site.xml.*

| Name | Value | Notes |
|---|---|---|
| `gpfs.sync.queue` | `True` | Use queues to merge commit logs in to a single write instead of multiple small writes.<br><br>Add the configuration through web GUI.<br><br>Choose **HBase** service **> Configs**, in the **Custom hbase-site** section, add this configuration and set it to the specific value. |
| `gpfs.sync.range` | `True` | Use the sync-range semantics for fsync to sync the log that was written instead of using fsync for data blocks.<br><br>Add the configuration through web GUI.<br><br>Choose **HBase** service **> Configs**, in the **Custom hbase-site** section, add this configuration and set it to the specific value. |
| `hbase.fsutil.hdfs.impl` | `org.apache.hadoop.hbase.gpfs.util.FSGPFSUtils` | Add the configuration through web GUI.<br><br>Choose **HBase** service **> Configs**, in the **Custom hbase-site** section, add this configuration and set it to the specific value. |
| `hbase.regionserver.hlog.writer.impl` | `org.apache.hadoop.hbase.gpfs.regionserver.wal.PreallocatedProtobufLogWriter` | Required for enhanced performance on the hbase log writes by pre-allocating space for log files.<br><br>Add the configuration through web GUI.<br><br>Choose **HBase** service **> Configs**, in the **Custom hbase-site** section, add this configuration and set it to the specific value. |
| `hbase.regionserver.hlog.reader.impl` | `org.apache.hadoop.hbase.gpfs.regionserver.wal.PreallocatedProtobufLogReader` | Required for enhanced performance on the hbase log reads by pre-allocating space for log files.<br><br>Add the configuration through web GUI.<br><br>Choose **HBase** service **> Configs**, in the **Custom hbase-site** section, add this configuration and set it to the specific value. |

For Spark, site-specific customization is done by specifying overriding parameters in the `Spark-site.xml` file.

If using IBM BigInsights, all changes must be made through IBM BigInsights Web GUI instead of modifying the configuration file directly.

You must stop **Spark** services before applying the following changes and start **Spark** service for the changes to take effect.

*Table 48. Modifications to Hadoop configuration—Spark-site.xml.*

| Name | Value | Notes |
|---|---|---|
| spark-env template | See notes | This value maintains Spark services running environment.<br><br>Update the configuration through web GUI.<br><br>Choose **Spark** service **> Advanced spark-env**, in the **spark-env template** section<br><br>Add **SPARK_CLASSPATH** parameter with the value:<br>`export SPARK_CLASSPATH= /usr/iop/current/hadoop-client/*: /usr/iop/current/hadoop-client/lib/*` |
| spark.sql.hive.metastore.sharedPrefixes | `org.apache.hadoop.fs.gpfs,org. apache.hadoop.hbase.gpfs,org.apache. hadoop.mapred.gpfs,org.apache.hadoop.pig.gpfs, org.apache.hadoop` | Update the configuration through web GUI.<br><br>Choose **Spark**″ service **> Configs > Custom spark-defaults**, add the variable with the specified value. |

### Propagating changes to all cluster nodes

- If you are using IBM BigInsights, after updating the applicable configurations through the web GUI, changes are propagated to the entire cluster.
- If you are using IBM Platform Symphony or another Hadoop distribution, follow the steps provided by the vendor to update all the configurations files like `core-site.xml` and `mapred-site.xml` files on all Hadoop cluster nodes.

## Upgrading IBM Spectrum Scale connector

When you are upgrading the connector for one node, the Hadoop applications running over the node cannot access the data in the IBM Spectrum Scale file system, although upgrading the connector does not impact the IBM Spectrum Scale file system service over the node. The customer needs to plan upgrading the connector according to the Hadoop application service.

### Upgrading IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.4, 4.1.0.5, 4.1.0.6, or 4.1.0.7 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.4, 4.1.0.5, 4.1.0.6, or 4.1.0.7 release, this section explains the steps required to upgrade the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

Also, you must ensure there are no two versions of `hadoop-gpfs-*.jar` and `libgpfshadoop.so` under the Hadoop distribution, to avoid exceptions when running jobs over Hadoop.

1. Remove any links or copies of the `hadoop-gpfs-2.4.jar` file from your Hadoop distribution directory. Also, remove any links or copies of the `libgpfshadoop.64.so` file from your Hadoop distribution directory.

   **Note:** For IBM BigInsights IOP 4.0, the distribution directory is `/usr/iop/4.0.0.0`.
2. Stop the current connector daemon:
   ```
   # ps -elf | grep gpfs-connector-daemon
   # kill -9 <pid-of-connector-daemon>
   ```
3. Run the following commands, to remove callbacks from IBM Spectrum Scale:
   ```
   cd /usr/lpp/mmfs/fpo/hadoop-2.4/install_script
   ./gpfs-callbacks.sh --delete
   ```

Run the **mmlscallbacks all** command to check whether connector-related callbacks, such as `callback ID start-connector-daemon` and `stop-connector-daemon`, are removed. The IBM Spectrum Scale Hadoop connector callbacks are cluster-wide and this step is required to be done over any one of nodes.

4. Remove the following files:

```
rm -f /var/mmfs/etc/gpfs-callbacks.sh
rm -f /var/mmfs/etc/gpfs-callback_start_connector_daemon.sh
rm -f /var/mmfs/etc/gpfs-callback_stop_connector_daemon.sh
rm -f /var/mmfs/etc/gpfs-connector-daemon
```

5. Run the following installation command:

```
–ivh gpfs.hadoop-connector-2.7.0-*.<platform-arch>.rpm
```

6. **mmhadoopctl connector attach --distribution BigInsights**

   **Note:** The **mmhadoopctl** command does not display any help in IBM Spectrum Scale 4.1.0.4, 4.1.0.5, 4.1.0.6, or 4.1.0.7 releases.

7. Restart the IBM BigInsights IOP services over the current operating node.

## Upgrading IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases, this section explains the steps required to upgrade the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

Also, ensure that there are no two versions of `hadoop-gpfs-*.jar` and `libgpfshadoop.so` under the Hadoop distribution, to avoid exceptions when running jobs over Hadoop.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. Run the following installation command:

```
rpm –ivh --force gpfs.hadoop-connector-2.7.0-<your-new-build-number>.<platform-arch>.rpm
```

   **Note:** : To avoid conflict for `/usr/lpp/mmfs/bin/mmhadoopctl`, `--force` is required in this step.

4. **mmhadoopctl connector attach --distribution BigInsights**
5. Restart the IBM BigInsights IOP services over the current operating node.

## Upgrading IBM Spectrum Scale Hadoop connector 2.4 or 2.5 over IBM Spectrum Scale 4.1.1 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 or 2.5 in IBM Spectrum Scale 4.1.1 and later, this section explains the steps required to upgrade the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

Also, ensure that there are no two versions of `hadoop-gpfs-*.jar` and `libgpfshadoop.so` under the Hadoop distribution, to avoid exceptions when running jobs over Hadoop.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. **rpm -e gpfs.hadoop-2-connector**
4. Run the following installation command:

```
rpm –ivh gpfs.hadoop-connector-2.7.0-*.<platform-arch>.rpm
```

5. **mmhadoopctl connector attach --distribution BigInsights**
6. Restart the IBM BigInsights IOP services over the current operating node.

### Upgrading IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases, this section explains the steps required to upgrade the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

Also, ensure that there are no two versions of `hadoop-gpfs-*.jar` and `libgpfshadoop.so` under the Hadoop distribution, to avoid exceptions when running jobs over Hadoop.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. Run the following installation command:
   **rpm –ivh --force gpfs.hadoop-connector-2.7.0-<your-new-build-number>.<platform-arch>.rpm**
4. **mmhadoopctl connector attach --distribution BigInsights**
5. Restart the IBM BigInsights IOP services over the current operating node.

### Upgrading IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7, 4.1.0.8, or 4.1.1+ releases

For users who are using IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7, 4.1.0.8, or 4.1.1+ releases, this section explains the steps required to upgrade the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

Also, ensure that there are no two versions of `hadoop-gpfs-*.jar` and `libgpfshadoop.so` under the Hadoop distribution, to avoid exceptions when running jobs over Hadoop.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. **rpm -e gpfs.hadoop-connector**
4. Run the following installation command:
   **rpm –ivh gpfs.hadoop-connector-2.7.0-<your-new-build-number>.<platform-arch>.rpm**
5. **mmhadoopctl connector attach --distribution BigInsights**
6. Restart the IBM BigInsights IOP services over the current operating node.

## Deploy IBM Spectrum Scale using Ambari

IBM Spectrum Scale supports deploying IBM BigInsights IOP 4.1 together with IBM Spectrum Scale through Ambari 2.1. Integration of IBM BigInsights IOP and IBM Spectrum Scale leads to an easy deployment of their Hadoop cluster using the GUI mode.

You can download the `gpfs.ambari` rpm package from the IBM developerWorks GPFS Wiki . For more information, see the *DeployBigInsights4.1_SpectrumScale_with_Ambari 2.1* guide in the IBM developerWorks GPFS Wiki.

# HDFS transparency

IBM Spectrum Scale HDFS transparency offers a set of interfaces that allows applications to use HDFS Client to access IBM Spectrum Scale through HDFS RPC requests.

All data transmission and metadata operations in HDFS are through the RPC mechanism and processed by the NameNode and the DataNode services within HDFS. IBM Spectrum Scale HDFS transparency implementation integrates both the NameNode and the DataNode services and responds to the request as in HDFS. Advantages of HDFS transparency are as follows:

- HDFS compliant APIs or shell-interface command.
- Application client isolation from storage. Application Client may access data in IBM Spectrum Scale without GPFS client installed.
- Improved security management by Kerberos authentication and encryption for RPCs.
- Simplified file system monitor by Hadoop Metrics2 integration.

# Supported IBM Spectrum Scale storage mode

## Local Storage Mode

HDFS transparency allows big data applications to access IBM Spectrum Scale local storage mode-File Placement Optimizer (FPO) mode, and enables the support for shared storage mode (such as SAN-based storage, ESS) starting with gpfs.hdfs-protocol.2.7.0-1 package.

In FPO mode, data blocks are stored in chunks in IBM Spectrum Scale, and replicated to protect against disk or node failure. DFS clients run over the storage node so it can leverage the data locality for executing the tasks quickly. In such a storage mode configuration, short-circuit read is recommended to improve the access efficiency.

## Shared Storage Mode

HDFS transparency allows big data applications to access data stored in shared storage mode (such SAN-based storage, IBM Elastic Storage™ Server).

In this mode, data is stored in shared storage systems which offers better storage efficiency than local storage. RAID and other technology can be used to protect hardware failure instead of using data replication.

DFS clients access data through the HDFS protocol remote procedure call (RPC). When a DFS Client requests to write blocks to IBM Spectrum Scale, the HDFS transparency NameNode selects DataNode randomly for this request. Especially, when DFS Client is on a DataNode, then that node is selected for this request. When the DFS Client requests to **getBlockLocation** of an existing block, NameNode selects a DataNodes randomly for this request. For example, if the **dfs.replication** parameter is set to 3, then 3 DataNodes are returned for a **getBlockLocation** request. If **dfs.replication** parameter is set to 1, then a single DataNode is returned for the **getBlockLocation** request.

HDFS transparency allows Hadoop application to access data stored in both local IBM Spectrum Scale file system and remote IBM Spectrum Scale file system from multiple cluster.

# Hadoop cluster planning

In an Hadoop cluster that runs the HDFS protocol, a node can take on the roles of DFS Client, a NameNode, or a DataNode or all of them. The Hadoop cluster might contain nodes that are all a part of an IBM Spectrum Scale cluster or it might contain some of the nodes in the IBM Spectrum Scale cluster.

## NameNode

You can specify a single NameNode or multiple NameNodes to protect against a single point of failure in the cluster. For more information, see “High availability configuration” on page 500. The NameNode must be selected from the IBM Spectrum Scale cluster and must have a robust configuration to reduce the chances of a single-node failure. The NameNode is defined by setting the `fs.defaultFS` parameter to the hostname of the NameNode in the `core-site.xml` file in Hadoop 2.4, 2.5, and 2.7 release.

**Note:** The Secondary NameNode in native HDFS is not needed for HDFS Transparency because the HDFS Transparency NameNode is stateless and does not maintain an FSImage or EditLog like state information.

## DataNode

You can specify multiple DataNodes in a cluster. The DataNodes must be a part of an IBM Spectrum Scale cluster. The DataNodes are specified by listing their hostnames in the `slaves` configuration file.

## DFS Client

DFS Client can be a part of an IBM Spectrum Scale cluster. When the DFS Client is a part of an IBM Spectrum Scale cluster, it can read data from IBM Spectrum Scale through RPC or use the short-circuit mode. Otherwise, the DFS Client can access data from IBM Spectrum Scale only through an RPC. You can specify the NameNode address in DFS Client configuration so that DFS Client can communicate with the appropriate NameNode service.

The purpose of cluster planning is to define the node roles: Hadoop node, HDFS transparency node, and GPFS node.

## Node roles planning

**Node roles planning in FPO mode:**

In FPO mode, all nodes are Target file not found (GPFS) nodes in FPO nodes, Hadoop nodes, and HDFS transparency nodes.

In this mode, Hadoop cluster must be larger than or equal to the HDFS transparency cluster. Hadoop cluster might be smaller than HDFS transparency cluster but this configuration is not typical and not recommended. Also, the HDFS transparency cluster must be smaller than or equal to IBM Spectrum Scale cluster because the HDFS transparency needs to read/write data from the local mounted file system. Usually, in FPO mode, the HDFS transparency cluster is equal to the IBM Spectrum Scale cluster.

**Note:** Some nodes in the IBM Spectrum Scale (GPFS) FPO cluster might be GPFS clients without any disks in the file system.

**Node roles planning in shared storage mode:**

If the IO stress is heavy, you might exclude the NSD servers from HDFS transparency cluster. Typically, in a shared storage mode, all nodes in Hadoop cluster might be GPFS client free, which means GPFS is not needed to be installed on these Hadoop nodes.

**Note:** All HDFS transparency nodes require GPFS to be installed and mounted.

For shared storage, it is recommended to deploy HDFS Transparency NameNode and DataNode services over nodes with local disk access path. This can reduce the amount of network traffic.

**Integration with Hadoop distributions:**

If you deploy HDFS transparency with a Hadoop distribution, such as IBM BigInsights IOP, you should configure the native HDFS NameNode as the HDFS Transparency NameNode and add this node into the IBM Spectrum Scale cluster. This setup will result in fewer configuration changes.

If the HDFS Transparency NameNode is not the same as the native HDFS NameNode, some services might fail to start and may require additional configuration changes.

## Hardware configuration

The following is the recommended configuration for Hadoop nodes:

- 10Gb network
- 100GB physical memory
- 10~20 internal SAS/SATA disks per node, and
- 8+ physical cores

## Hadoop service roles

In a Hadoop ecosystem, there are a lot of different roles for different components. For example, HBase Master Server, Yarn Resource Manager, Yarn Node Manager.

You need to plan to spread these master roles over different nodes as evenly as possible. If you put all these master roles onto a single node, the memory might become an issue.

When running Hadoop over IBM Spectrum Scale, it is recommended that up to 25% of the physical memory be reserved for the GPFS pagepool with a maximum of 16GB. If HBase is being used, it is recommended that up to 30% of the physical memory be reserved for the GPFS pagepool. If the node has less than 100GB of physical memory, then the heap size for Hadoop Master services needs to be carefully planned. If HDFS transparency NameNode service and HBase Master service are resident on the same physical node, HBase workload stress may result in Out of Memory (OOM) exceptions.

## Dual network interfaces

The following section is only applicable for IBM Spectrum Scale FPO (local storage) mode, and does not impact Hadoop clusters running over a shared storage configuration (for example, SAN-based cluster, or ESS).

If the FPO cluster has a dual 10 Gb network, you have two configuration options:

- The first option is to bond the two network interfaces and deploy the IBM Spectrum Scale cluster and the Hadoop cluster over the bonded interface.
- The second option is to configure one network interface for the Hadoop services including the HDFS transparency service and configure the other network interface for IBM Spectrum Scale to use for data traffic. This configuration can minimize interference between disk I/O and application communication.

For the second option we recommend the following steps to ensure that the Hadoop applications can exploit data locality for better performance:

1. Configure the first network interface with one subnet address (for example, 192.168.1.0). Configure the second network interface as another subnet address (for example, 192.168.2.0).
2. Create the IBM Spectrum Scale cluster and NSDs with the IP or hostname from the first network interface.
3. Install the Hadoop cluster and HDFS transparency services by using the IP addresses or hostnames from the first network interface.
4. Run `mmchconfig subnets=192.168.2.0 -N all`.

**Note:** 192.168.2.0 is the subnet used for IBM Spectrum Scale data traffic.

For Hadoop map/reduce jobs, the scheduler Yarn checks the block location. HDFS Transparency returns the hostname (which is used to create IBM Spectrum Scale cluster) as block location to Yarn. Yarn then checks the hostname within the NodeManager host list. If Yarn cannot find the hostname within the NodeManager list, Yarn cannot schedule the tasks according to data locality. The suggested configuration can ensure that the hostname for block location can be found in Yarn's NodeManager list and therefore Yarn can schedule the task according to data locality.

For a Hadoop distribution like IBM BigInsights IOP, all Hadoop components are managed by Ambari™. All Hadoop components, HDFS transparency, and IBM Spectrum Scale cluster must be created using one network interface and use the second network interface for GPFS data traffic.

## Installation and configuration of HDFS transparency

This section describes the installation and configuration of HDFS transparency with IBM Spectrum Scale.

### Installation of HDFS transparency

IBM Spectrum Scale HDFS transparency must be installed on nodes that serve as NameNodes or DataNodes.

HDFS Transparency works with IBM Spectrum Scale 4.1 or later. For Linux distro version, see IBM Spectrum Scale FAQ in IBM Knowledge Center.

Use the following command to install the HDFS transparency RPM:

```
# rpm -hiv gpfs.hdfs-protocol-2.7.0-0.x86_64.rpm
```

This package has the following dependencies:

- `libacl`
- `libattr`
- `openjdk 7.0+`

HDFS transparency files are installed under the `/usr/lpp/mmfs/hadoop` directory. To list the contents of this directory, use the following command:

```
#ls /usr/lpp/mmfs/hadoop/
bin etc lib libexec license logs README run sbin share
```

The following directories can be added to the system shell path for convenience:

- `/usr/lpp/mmfs/hadoop/bin`
- `/usr/lpp/mmfs/hadoop/sbin`

### Configuration of HDFS transparency

For configuring, you must install Hadoop distribution under `$YOUR_HADOOP_PREFIX` on each machine in the cluster. The configurations for IBM Spectrum Scale HDFS transparency are located under `/usr/lpp/mmfs/hadoop/etc/hadoop` for any Hadoop distribution. Configurations for Hadoop distribution are located in different locations, for example, `/etc/hadoop/conf` for IBM BigInsights IOP.

The `core-site.xml` and `hdfs-site.xml` configuration files should be synced and kept identical for the IBM Spectrum Scale HDFS transparency and Hadoop distribution. The `log4j.properties` configuration file can differ between the IBM Spectrum Scale HDFS transparency and the native Hadoop distribution.

**OS tuning for all nodes:**

**ulimit tuning**

For all nodes, `ulimit -n` and `ulimit -u` must be larger than or equal to 65536. Smaller value makes Hadoop java processes report unexpected exceptions.

In Red Hat, add the following lines at the end of `/etc/security/limits.conf` file:

```
*       soft nofile 65536
*       hard nofile 65536

*       soft nproc 65536
*       hard nproc 65536
```

For other Linux systems, you might check the linux guide of your distro.

After the above change, you need to restart all services to make it effective.

**Note:** This should be done on all nodes including the Hadoop client nodes and the HDFS Transparency nodes.

**Configure Hadoop Nodes:**

For configuring Hadoop nodes, the following files need to be updated for open source Apache Hadoop:

- `core-site.xml`
- slaves

In this example, hostname of NameNode service is `hs22n44`. Edit the following files for a standard Hadoop configuration.

In `$HADOOP_PREFIX/etc/hadoop/core-site.xml`, ensure that the configuration is

```
fs.defaultFS:
<property>
<name>fs.defaultFS</name>
<value>hdfs://hs22n44:9000</value>
</property>
```

Replace `hs22n44:9000` with the hostname of your NameNode service and preferred port number.

User can customize other configuration parameters like service ports. For more information, see Welcome to Apache™ Hadoop®!.

In `$YOUR_HADOOP_PREFIX/etc/hadoop/slaves` file, ensure that all the DataNodes are listed in the file. For example:

```
# cat $HADOOP_PREFIX/etc/hadoop/slaves
hs22n44
hs22n54
hs22n45
```

As for `hdfs-site.xml` and other detailed configuration settings in `core-site.xml`, follow the link Welcome to Apache™ Hadoop®! to configure Hadoop nodes. So far, the following must be configured to avoid unexpected exceptions from Hadoop:

```
<property>
<name>dfs.datanode.handler.count</name>
<value>40</value>
</property>

<property>
```

```
<name>dfs.datanode.handler.count</name>
<value>400</value>
</property>

<property>
<name>dfs.datanode.max.transfer.threads</name>
<value>8192</value>
</property>
```

After the configuration, sync them to all Hadoop nodes.

**Note:** If you take Hadoop distribution, like IBM BigInsights, you need to configure Hadoop components (for example. HBase, Hive, oozie, etc) in the management GUI, for example, Ambari for IBM BigInsights.

Also, do not export the environmental variables *HADOOP_HDFS_HOME*, *HADOOP_MAPRED_HOME*, *HADOOP_COMMON_HOME*, *HADOOP_COMMON_LIB_NATIVE_DIR* on the HDFS Transparency nodes as this can cause issues.

**Configure HDFS transparency nodes:**

*Sync Hadoop configurations:*

By default, HDFS transparency uses the `core-site.xml` and `hdfs-site.xml` configuration files from the Hadoop distribution, along with the `gpfs-site.xml` located under `/usr/lpp/mmfs/hadoop/etc/hadoop` directory.

The following configuraion files need to be distributed to all the HDFS transparency nodes:

- `core-site.xml`
- `hdfs-site.xml`
- `slaves`
- `log4j.properties`

If the HDFS Transparency nodes are also running Hadoop, you can use the following command to sync the configuration files to the rest of the HDFS Transparency nodes. Run this command on one of the HDFS Transparency nodes (e.g. hdfs_transparency_node1) after ensuring that the HDFS Transparency service is running:

```
hdfs_transparency_node1#/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf <hadoop-conf-dir>
```

If the HDFS transparency nodes are not running Hadoop, use a tool like scp (secure copy) to distribute the following files to the `/usr/lpp/mmfs/hadoop/etc/hadoop/` directory on all the HDFS transparency nodes:

- `<hadoop-conf-dir>/core-site.xml`
- `<hadoop-conf-dir>/hdfs-site.xml`
- `<hadoop-conf-dir>/log4j.properties`

*Configure the storage mode:*

Modify the `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml` file on the `hdfs_transparency_node1` node:

```
<property>
<name>gpfs.storage.type</name>
<value>local</value>
</property>
```

The property *gpfs.storage.type* is used to specify storage mode. The storage mode can be local or shared. This is a required configuration parameter and the `gpfs-site.xml` file must be synced to all the HDFS transparency nodes after the above modification.

*Update other configuration files:*

**Note:** To configure Hadoop HDFS, Yarn, etc refer to the `hadoop.apache.org` website. This section focuses on HDFS configuration parameters.

**Configurations for Apache Hadoop**

Modify the `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml` file on the `hdfs_transparency_node1` node:

```
<property>
<name>gpfs.mnt.dir</name>
<value>/gpfs_mount_point</value>
</property>

<property>
<name>gpfs.data.dir</name>
<value>data_dir</value>
</property>

<property>
<name>gpfs.supergroup</name>
<value>hadoop</value>
</property>

<property>
<name>gpfs.replica.enforced</name>
<value>dfs</value>
</property>
```

In `gpfs-site.xml`, all Hadoop data is stored under `/gpfs_mount_point/data_dir` directory. You can have two Hadoop clusters over the same file system and these clusters are isolated from each other. When Hadoop operates the file, one limitation is that if there is a link under the `/gpfs_mount_point/data_dir` directory that points to a file outside the `/gpfs_mount_point/data_dir` directory, it reports an exception because that file is not accessible by Hadoop.

If you do not want to explicitly configure the **`gpfs.data.dir`** parameter, you can leave it as null. For example, keep its value as <value></value>.

**Note:** Do not configure it as <value>/</value>.

The `gpf.supergroup` must be configured according to your cluster. You need to add some Hadoop users, such as HDFS, yarn, hbase, hive, oozie, etc under the same group named Hadoop and configure `gpfs.supergroup` as Hadoop. You might specify two or more comma-separated groups as `gpfs.supergroup`. For example, `group1,group2,group3`.

**Note:** Users in `gpfs.supergroup` are super users and they can control all the data in `/gpfs_mount_point/data_dir` directory. This is similar to the user root in Linux.

The **`gpfs.replica.enforced`** parameter is used to control the replica rules. Hadoop controls the data replication through the **`dfs.replication`** parameter. When running Hadoop over IBM Spectrum Scale, IBM Spectrum Scale has its own replication rules. If you configure `gpfs.replica.enforced` as dfs , then **`dfs.replication`** is always effective unless you specify **`dfs.replication`** in the command options when submitting jobs. If `gpfs.replica.enforced` is set to gpfs, then all data will be replicated according to IBM Spectrum Scale configuration settings. The default value for this parameter is set to dfs.

Usually, you must not change `core-site.xml` and `hdfs-site.xml` located under `/usr/lpp/mmfs/hadoop/etc/hadoop/`. These two files must be consistent as the files used by Hadoop nodes.

You need to modify `/usr/lpp/mmfs/hadoop/etc/hadoop/slaves` to add all HDFS transparency DataNode hostnames and one hostname per line, for example:

```
# cat /usr/lpp/mmfs/hadoop/etc/hadoop/slaves
hs22n44
hs22n54
hs22n45
```

You might check /usr/lpp/mmfs/hadoop/etc/hadoop/log4j.properties and modify it accordingly. This file might be different as the log4j.properties used by Hadoop nodes.

After you finish the configurations, use the following command to sync it to all IBM Spectrum Scale HDFS transparency nodes:

```
hdfs_transparency_node1#/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop
```

**Configuration for IBM BigInsights IOP**

In IBM BigInsights IOP 4.0/4.1, IOP and IBM Spectrum Scale HDFS transparency are integrated manually. Therefore, it is easy to configure the IBM Spectrum Scale HDFS transparency. For IBM BigInsights IOP 4.1, if you deployed IOP 4.1 with IBM Spectrum Scale Ambari integration, you can email scale@us.ibm.com for more information. If you deployed IOP 4.1 without IBM Spectrum Scale Ambari integration, perform the following steps:

1. On the node hdfs_transparency_node1, run the following command to sync IBM BigInsights IOP configuration into IBM Spectrum Scale HDFS transparency configuration directory: **/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf /etc/hadoop/conf/**
2. On the node hdfs_transparency_node1, refer the Configurations for Apache Hadoop section to create the /usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml file, update the /usr/lpp/mmfs/hadoop/etc/hadoop/slaves and /usr/lpp/mmfs/hadoop/etc/hadoop/log4j.properties files.
3. On the node hdfs_transparency_node1, run the **/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf /usr/lpp/mmfs/ hadoop/etc/hadoop/** command to sync the gpfs-site.xml, core-site.xml, hdfs-site.xml, slaves and log4j.properties to all the IBM Spectrum Scale HDFS transparency nodes.

*Update environment variables for HDFS transparency service:*

The administrator might need to update some environment variables for the HDFS Transparency service. For example, change JVM options or Hadoop environment variables like **HADOOP_LOG_DIR**.

In order to update this, follow these steps:

1. On the HDFS Transparency NameNode, modify the /usr/lpp/mmfs/Hadoop/etc/hadoop/hadoop-env.sh and other files as necessary.
2. Sync the changes to all the HDFS Transparency nodes by executing the following command:
   ```
   #/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop
   ```

## Start and stop the service

To start and stop HDFS transparency services, you need to be a root user. You also need to keep native HDFS service down because HDFS transparency provides the same services. If you keep both services up, it reports conflict in service network port number. You need to restart all other Hadoop services, such as Yarn, Hive, HBase, etc after you replace native HDFS with HDFS transparency.

To start the HDFS transparency service on the NameNode, use the following command:

```
/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector start
```

To stop the HDFS transparency service on the NameNode, use the following command:

```
/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector stop
```

### Health check the service

Any user can conduct a health check of the Hadoop service.

To conduct a health check of the service, use the following command:

```
/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector getstate
```

## High availability configuration

### Manual HA switch configuration

The high availability (HA) implementation follows the HDFS High Availability feature by using NFS. You can define a GPFS directory instead of an NFS directory to synchronize information between two NameNodes.

In the following configuration example, the HDFS `nameservice` ID is `mycluster` and `name node` IDs are `nn1` and `nn2`. Configuration must be done in the `core-site.xml` file by defining `fs.defaultFS` with the `nameservice` ID.

1. Define the nameservice ID in the `core-site.xml` file that is used by the Hadoop distribution. If you are using IBM BigInsights IOP, change this configuration in the Ambari GUI and restart the HDFS services to synchronize it with all the Hadoop nodes.

   ```
   <property>
   <name>fs.defaultFS</name>
   <value>hdfs://mycluster</value>
   </property>
   ```

2. Configure the `hdfs-site.xml` file that is used by the Hadoop distribution. If you are using IBM BigInsights IOP, change these configurations in the Ambari GUI and restart the HDFS services to synchronize it with all the Hadoop nodes.

   ```
   <property>
   <!--define dfs.nameservices ID-->
   <name>dfs.nameservices</name>
   <value>mycluster</value>
   </property>

   <property>
   <!--define name nodes ID for HA-->
   <name>dfs.ha.namenodes.mycluster</name>
   <value>nn1,nn2</value>
   </property>

   <property>
   <!--Actual hostname and rpc address of name node ID-->
   <name>dfs.namenode.rpc-address.mycluster.nn1</name>
   <value>c8f2n06.gpfs.net:8020</value>
   </property>

   <property>
   <!--Actual hostname and rpc address of name node ID-->
   <name>dfs.namenode.rpc-address.mycluster.nn2</name>
   <value>c8f2n07.gpfs.net:8020</value>
   </property>

   <property>
   <!--Actual hostname and http address of name node ID-->
   <name>dfs.namenode.http-address.mycluster.nn1</name>
   <value>c8f2n06.gpfs.net:50070</value>
   </property>

   <property>
   <!--Actual hostname and http address of name node ID-->
   <name>dfs.namenode.http-address.mycluster.nn2</name>
   <value>c8f2n07.gpfs.net:50070</value>
   </property>
   ```

```
<property>
<!--Shared directory used for status sync up-->
<name>dfs.namenode.shared.edits.dir</name>
<value>/<gpfs.mnt.dir>/<gpfs.data.dir>/HA</value>
</property>

<property>
<name>dfs.ha.standby.checkpoints</name>
<value>false</value>
</property>

<property>
<name>dfs.client.failover.proxy.provider.mycluster</name>
<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
```

The **dfs.namenode.shared.edits.dir** configuration parameter must be consistent with `gpfs.mnt.dir` and `gpfs.data.dir` defined in `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml`. You can create the directory `/<gpfs.mnt.dir>/<gpfs.data.dir>/HA` and change the ownership to hdfs:hadoop before starting the HDFS transparency services.

The `dfs.ha.standby.checkpoints` must be set to `false`. Otherwise, you will see a log of exceptions in the standby NameNode logs, such as:

```
ERROR ha.StandbyCheckpointer (StandbyCheckpointer.java:doWork(371)) - Exception in doCheckpoint
```

In HDFS transparency, NameNode does not maintain a state such as fsImage or editLogs as in native HDFS. Therefore, there is no need to perform checkpoints from the standby NameNode service.

The **dfs.client.failover.proxy.provider.mycluster** configuration parameter must be changed according to the name service ID. In the above example, the name service ID is configured as mycluster in `core-site.xml`. Therefore, the configuration name is `dfs.client.failover.proxy.provider.mycluster`.

**Note:** If you enable Short Circuit Read in the Short Circuit Read Configuration section, the value of the configuration parameter must be **org.apache.hadoop.gpfs.server.namenode.ha.ConfiguredFailoverProxyProvider**.

3. Follow the guide in the Sync Hadoop configurations section to synchronize `core-site.xml` and `hdfs-site.xml` from the Hadoop distribution to any one node that is running HDFS transparency services. For example, HDFS_Transparency_node1.
4. On **HDFS_Transparency_node1**, modify `/usr/lpp/mmfs/hadoop/etc/hadoop/hdfs-site.xml`:

   ```
    <property>
   <name>dfs.client.failover.proxy.provider.mycluster</name>
   <value>org.apache.hadoop.gpfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
   </property>
   ```

   With this configuration, WebHDFS service functions correctly when NameNode HA is enabled.

   **Note:** On HDFS transparency nodes, the configuration value of the key **dfs.client.failover.proxy.provider.mycluster** in `hdfs-site.xml` is different from that in Step2.
5. On **HDFS_Transparency_node1**, run the command as the root user to synchronize the HDFS Transparency configuration to all the HDFS transparency nodes:

   **# mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop**
6. Start the HDFS transparency service by running the **mmhadoopctl** command:

   **# mmhadoopctl connector start**
7. After the service starts, both NameNodes are in the standby mode by default. You can activate one NameNode by using the following command so that it responds to the client:

   **# /usr/lpp/mmfs/hadoop/bin/gpfs haadmin -transitionToActive --forceactive [name node ID]**For example, you can activate the nn1 NameNode by running the following command:

**# /usr/lpp/mmfs/hadoop/bin/gpfs haadmin -transitionToActive -forceactive nn1**If the nn1 NameNode fails, you can activate another NameNode and relay the service by running the following command:

**# /usr/lpp/mmfs/hadoop/bin/gpfs haadmin -transitionToActive -forceactive nn2**

**Note:** The switch must be done manually. Automatic switch will be supported in the future releases. Use the following command to view the status of the NameNode:

**# /usr/lpp/mmfs/hadoop/bin/gpfs haadmin -getServiceState [name node ID]** After one NameNode becomes active, you can start the other Hadoop components, such as hbase and hive and run your Hadoop jobs.

**Note:** When HA is enabled for HDFS transparency, you might see the following exception in the logs: Get corrupt file blocks returned error: Operation category READ is not supported in state standby.

These are known HDFS issues: HDFS-3447 and HDFS-8910.

## Automatic NameNode service HA

Automatic NameNode Service HA is supported in gpfs.hdfs-protocol 2.7.0-2 and later. The implementation of high availability (HA) is the same as NFS-based HA in native HDFS. The only difference is that except for the NFS shared directory in native HDFS, HA is not needed for HDFS transparency.

The prerequisite to configure automatic NameNode HA is to have zookeeper services running in the cluster.

**Configuring Automatic NameNode Service HA:**

If you take a Hadoop distro, such as IBM BigInsights IOP, the zookeeper service is deployed by default. However, if you select open-source Apache Hadoop, you must set up the zookeeper service by following the instruction on the zookeeper website.

**Note:** In the following configuration example, HDFS Transparency NameNode service ID is mycluster and NameNode IDs are nn1 and nn2. ZooKeeper server zk1.gpfs.net, zk2.gpfs.net and zk3.gpfs.net are configured to support automatic NameNode HA. The ZooKeeper servers must be started before starting the HDFS Transparency cluster.

1. Define the NameNode service ID in the core-site.xml that is used by your Hadoop distribution.

   **Note:** If you are using IBM BigInsights IOP, you can change this configuration in Ambari GUI and restart the HDFS services to synchronize it with all the Hadoop nodes.

   ```
   <property>
   <name>fs.defaultFS</name>
   <value>hdfs://mycluster</value>
   </property>
   ```

2. Configure the hdfs-site.xml file used by your Hadoop distribution:

   **Note:** If you are using IBM BigInsights IOP, you can change this configuration in Ambari GUI and restart the HDFS services to synchronize it with all the Hadoop nodes.

   ```
   <property>
   <!--define dfs.nameservices ID-->
   <name>dfs.nameservices</name>
   <value>mycluster</value>
   </property>

   <property>
   <!--define name nodes ID for HA-->
   <name>dfs.ha.namenodes.mycluster</name>
   <value>nn1,nn2</value>
   </property>
   ```

```
<property>
<!--Actual hostname and rpc address of name node ID-->
<name>dfs.namenode.rpc-address.mycluster.nn1</name>
<value>c8f2n06.gpfs.net:8020</value>
</property>

<property>
<!--Actual hostname and rpc address of name node ID-->
<name>dfs.namenode.rpc-address.mycluster.nn2</name>
<value>c8f2n07.gpfs.net:8020</value>
</property>

<property>
<!--Actual hostname and http address of name node ID-->
<name>dfs.namenode.http-address.mycluster.nn1</name>
<value>c8f2n06.gpfs.net:50070</value>
</property>

<property>
<!--Actual hostname and http address of name node ID-->
<name>dfs.namenode.http-address.mycluster.nn2</name>
<value>c8f2n07.gpfs.net:50070</value>
</property>

<property>
<!--Shared directory used for status sync up-->
<name>dfs.namenode.shared.edits.dir</name>
<value>/<gpfs.mnt.dir>/<gpfs.data.dir>/HA</value>
</property>

<property>
<name>dfs.ha.standby.checkpoints</name>
<value>false</value>
</property>

<property>
<name>dfs.client.failover.proxy.provider.mycluster</name>
<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>

<property>
<name>dfs.ha.fencing.methods</name>
<value>shell(/bin/true)</value>
</property>

<property>
<name>dfs.ha.automatic-failover.enabled</name>
<value>true</value>
</property>

<property>
<name>ha.zookeeper.quorum</name>
<value>zk1.gpfs.net:2181,zk2.gpfs.net:2181,zk3.gpfs.net:2181</value>
</property>
```

The configuration **dfs.namenode.shared.edits.dir** should be consistent with `gpfs.mnt.dir` and `gpfs.data.dir` defined in `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml`. You could create the directory `/<gpfs.mnt.dir>/<gpfs.data.dir>/HA` and make it owned by hdfs:hadoop before starting HDFS transparency services.

The **dfs.ha.standby.checkpoints** should be set as false. If not, you will see a log of exceptions in the standby NameNode logs. For example,

```
ERROR ha.StandbyCheckpointer (StandbyCheckpointer.java:doWork(371)) - Exception in doCheckpoint.
```

HDFS transparency does not have fsImage and editLogs. Therefore, do not perform checkpoints from the standby NameNode service.

The configuration name dfs.client.failover.proxy.provider.mycluster must be changed according to the nameservice ID. In the above example, the nameservice ID is configured as *mycluster* in core-site.xml. Therefore, the configuration name is `dfs.client.failover.proxy.provider.mycluster`.

**Note:** If you enable Short Circuit Read, the value of this configuration must be *org.apache.hadoop.gpfs.server.namenode.ha.ConfiguredFailoverProxyProvider*.

3. To synchronize core-site.xml with hdfs-site.xml from your Hadoop distribution to any one node that is running HDFS transparency services, see Sync Hadoop configurations.
4. On HDFS_Transparency_node1, modify the `/usr/lpp/mmfs/hadoop/etc/hadoop/hdfs-site.xml`:

   ```
   <property>
   <name>dfs.client.failover.proxy.provider.mycluster</name>
   <value>org.apache.hadoop.gpfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
   </property>
   ```

   The WebHDFS service functions properly when NameNode HA is enabled.

   **Note:** On HDFS transparency nodes, the above configuration value in hdfs-site.xml is different as that in Step2.
5. On HDFS_Transparency_node1, run the following command as the root user to synchronize HDFS Transparency configuration with all HDFS transparency nodes:

   ```
   mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop
   ```
6. Start the HDFS Transparency service by running the **mmhadoopctl** command:

   ```
   mmhadoopctl connector start
   ```
7. Start the zkfc daemon:

   ```
   /usr/lpp/mmfs/hadoop/sbin/hadoop-daemon.sh start zkfc -formatZK
   ```

   You can remove the `formatZK` option. Run jps on the nn1 and nn2 name nodes to check if the DFSZKFailoverController process has been started.

   **Note:** If the `-formatZK` option is not added, the system displays the following exception: `FATAL org.apache.hadoop.ha.ZKFailoverController: Unable to start failover controller. Parent znode does not exist`
8. Run the following command to check that all NameNode services and DataNode services are functioning:

   ```
   mmhadoopctl connector getstate
   ```
9. Run the following command to check the state of NameNode services:

   ```
   /usr/lpp/mmfs/hadoop/bin/gpfs haadmin -getServiceState [name node ID]
   ```

   **Note:** When HA is enabled for HDFS transparency, the following exception might be logged: `Get corrupt file blocks returned error: Operation category READ is not supported in state standby.` These are unfixed HDFS issues: HDFS-3447 and HDFS-8910.

## Short-circuit read configuration

In HDFS, read requests go through the DataNode. When the client asks the DataNode to read a file, the DataNode reads that file off the disk and sends the data to the client over a TCP socket. The short-circuit read obtains the file descriptor from the DataNode, allowing the client to read the file directly.

This is possible only in cases where the client is co-located with the data and used in the FPO mode. Short-circuit reads provide a substantial performance boost to many applications.

**Note:** Short-circuit local reads can only be enabled on Hadoop 2.7.0. For more information on how to enable short-circuit reads on other Hadoop versions, contact `scale@us.ibm.com`.

## Configuring short-circuit local read

To configure short-circuit local reads, you need to enable `libhadoop.so` and use the DFS Client shipped by the IBM Spectrum Scale HDFS transparency, the package name is `gpfs.hdfs-protocol`. You cannot use standard HDFS DFS Client to enable the short-circuit mode over the HDFS transparency.

To enable `libhadoop.so`, compile the native library on the target machine or use the library shipped by IBM Spectrum Scale HDFS transparency. To compile the native library on the specific machine, do the following steps:

1. Download Hadoop source code from Hadoop community
2. Build by `mvn`: **`$ mvn package -Pdist,native -DskipTests -Dtar`**
3. Copy `hadoop-dist/target/hadoop-2.7.1/lib/native/libhadoop.so.*` to `$HADOOP_PREFIX/lib/native/`

   Or, to use the `libhadoop.so` delivered by the HDFS transparency, copy `/usr/lpp/mmfs/hadoop/lib/native/libhadoop.so` to `$HADOOP_PREFIX /lib/native/libhadoop.so`

   The shipped `libhadoop.so` is built on `x86_64`, `ppc64`, or `ppc64le` respectively.

   **Note:** This step must be done over all nodes running the Hadoop tasks.

**Enabling DFS Client:**
To enable DFS Client, shipped along with the HDFS transparency, on each node that accesses IBM Spectrum Scale in the short-circuit mode:

1. Back up **`hadoop-hdfs-2.7.0.jar`** using `$ mv $HADOOP_PREFIX/share/hadoop/hdfs/hadoop-hdfs-2.7.0.jar $HADOOP_PREFIX/share/hadoop/hdfs/hadoop-hdfs-2.7.0.jar.backup`
2. Link **`hadoop-gpfs-2.7.0.jar`** to classpath `$ln -s /usr/lpp/mmfs/hadoop/share/hadoop/hdfs/hadoop-gpfs-2.7.0.jar $HADOOP_PREFIX/share/hadoop/hdfs/hadoop-gpfs-2.7.0.jar`
3. Update the `core-site.xml` file with the following information:

```
<property>
  <name>fs.hdfs.impl</name>
  <value>org.apache.hadoop.gpfs.DistributedFileSystem</value>
</property>
```

Short-circuit reads make use of a UNIX domain socket. This is a special path in the file system that allows the client and the DataNodes to communicate. You need to set a path to this socket. The DataNode needs to be able to create this path. However, it must not be possible for any user except the HDFS user or root to create this path. Therefore, paths under `/var/run` or `/var/lib` folders are often used.

The client and the DataNode exchange information through a shared memory segment on the `/dev/shm` path. Short-circuit local reads need to be configured on both the DataNode and the client. Here is an example configuration.

```
<configuration>
<property>
<name>dfs.client.read.shortcircuit</name>
<value>true</value>
</property>
<property>
<name>dfs.domain.socket.path</name>
<value>/var/lib/hadoop-hdfs/dn_socket</value>
</property>
</configuration>
```

Sync up all these changes in the entire cluster and if needed, restart the service.

**Note:** The `/var/lib/hadoop-hdfs` and `dfs.domain.socket.path` must be created manually by the root user before running the short-circuit read. The `/var/lib/hadoop-hdfs` must be owned by the root user. If not, the DataNode service fails when starting up.

```
#mkdir -p  /var/lib/hadoop-hdfs
#chown root:root /var/lib/hadoop-hdfs
#touch   /var/lib/hadoop-hdfs/${dfs.dome.socket.path}
#chmod 666 /var/lib/hadoop-hdfs/${dfs.dome.socket.path}
```

The permission control in short-circuit reads is similar to the common user access in HDFS. If you have the permission to read the file, then you can access it through short-circuit read.

## Multiple Hadoop clusters over the same file system

By using HDFS transparency, you can configure multiple Hadoop clusters over the same IBM Spectrum Scale file system. For each Hadoop cluster, you need one HDFS transparency cluster to provide the filesystem service.

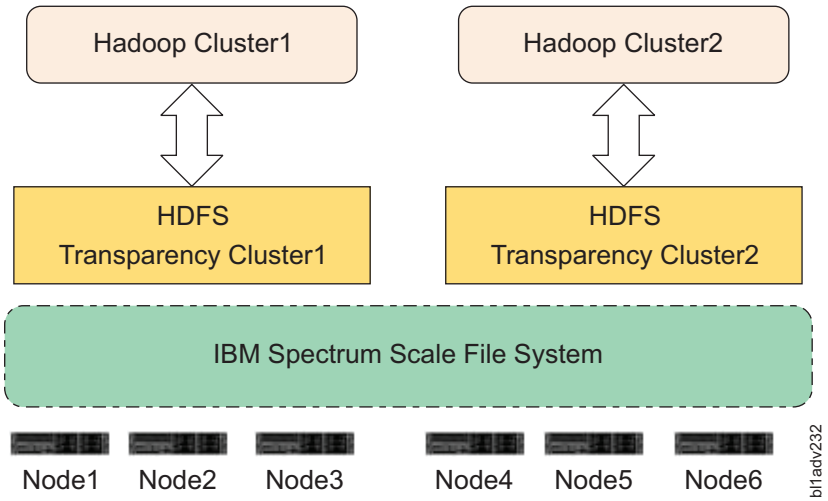



*Figure 18. Two Hadoop Clusters over the same IBM Spectrum Scale file system*

You can configure Node1 to Node6 as an IBM Spectrum Scale cluster (FPO or shared storage mode). Then configure Node1 to Node3 as one HDFS transparency cluster and Node4 to Node6 as another HDFS transparency cluster. HDFS transparency cluster1 and HDFS transparency cluster2 take different configurations by changing `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml`:

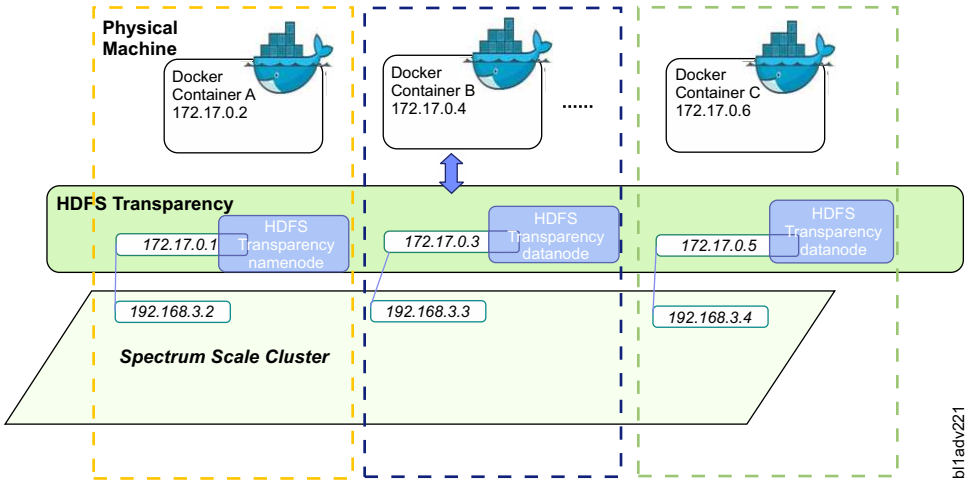
1. Change the gpfs-site.xml for HDFS transparency cluster1 to store the data under `/<gpfs-mount-point>/<hadoop1>` (gpfs.data.dir=hadoop1 in gpfs-site.xml).
2. Run **`mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop`** to synchronize the gpfs-site.xml from Step1 to all other nodes in HDFS transparency cluster1.
3. Change the gpfs-site.xml for HDFS transparency cluster2 to store the data under `/<gpfs-mount-point>/<hadoop2>` (gpfs.data.dir=hadoop2 in gpfs-site.xml).
4. Run `mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop` to synchronize the gpfs-site.xml from Step3 to all other nodes in HDFS transparency cluster2.
5. Restart the HDFS transparency services.

## Docker support

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

See the Docker website for Docker technology.

With HDFS transparency, you can run Hadoop Map/Reduce jobs in Docker and take IBM Spectrum Scale as the uniform data storage layer over the physical machines.




You can configure different Docker instances from different physical machines as one Hadoop cluster and run Map/Reduce jobs on the virtual Hadoop clusters. All Hadoop data is stored in the IBM Spectrum Scale file system over the physical machines. The `172.17.0.x` IP address over each physical machine is a network bridge adapter used for network communication among Docker instances from different physical machines. HDFS transparency services must be configured to monitor the network bridge and process the requests from Docker instances. After receiving the requests from Hadoop jobs running in Docker instances, HDFS transparency handles the I/O requests for the mounted IBM Spectrum Scale file system on the node.

## Configuring the Docker instance and HDFS transparency

1. Docker (version 1.9+) requires Redhat7+. Modify the Redhat Yum Repos to upgrade the selinux-policy and device-mapper-libs by running the following commands:
   - **`yum upgrade selinux-policy`**
   - **`yum upgrade device-mapper-libs`**
2. To install Docker engine (version 1.9+), See link.
3. Configure the network bridge adapter on physical machines. There can be only one network bridge adapter on one machine.

   **Note:** These configurations must be changed under `/etc/sysconfig/network-scripts/`:

```
[root@c3m3n04 network-scripts]# cat ifcfg-br0
DEVICE=br0
TYPE=Bridge
BOOTRPOTO=static
IPADDR=172.17.0.1
NETMASK=255.255.255.0
ONBOOT=yes

[root@c3m3n04 network-scripts]# cat ifcfg-enp11s0f0
# Generated by dracut initrd
DEVICE="enp11s0f0"
ONBOOT=yes
NETBOOT=yes
UUID="ca481ab0-4cdf-482e-b5d3-82be13a7621c"
IPV6INIT=yes
BOOTPROTO=static
HWADDR="e4:1f:13:be:5c:28"
TYPE=Ethernet
```

```
NAME="enp11s0f0"
IPADDR=192.168.3.2
BROADCAST=192.168.255.255
NETMASK=255.255.255.0
```

**Note:** You must modify the IPADDR, BROADCAST, and NETMASK according to your network configuration.

In this example, the br0 bridge adapter is bundled with the enp11s0f0 physical adapter. You must modify the code in the example for all the physical machines on which the Docker instances must be run.

4. Modify the Docker service script and start the Docker engine daemons on each node:

```
vim /usr/lib/systemd/system/docker.service
ExecStart=/usr/bin/docker daemon -b br0 -H fd://

service docker stop
service docker start
```

5. Configure the network route table on each machine:

```
route add -net 172.17.1.0/24 gw <replace-physical-node-ip-here> dev enp11s0f0
```

where <replace-physical-node-ip-here> is the IP address of your machine.

6. The IP addresses of the nodes must be different so that the Docker instances from one physical node can access the Docker instances in another physical node. Check if you can connect to the br0 IP address from another node.
7. Configure HDFS transparency and start the HDFS transparency services. Modify `/usr/lpp/mmfs/hadoop/etc/hadoop/core-site.xlm` and `/usr/lpp/mmfs/hadoop/etc/hadoop/slaves`. You must select the IP address from Docker network bridge adapter. Pull the Hadoop Docker image on each node:

```
docker pull sequenceiq/hadoop-docker:2.7.0
```

**Note:** We have selected the Hadoop Docker image from sequenceiq.

8. Start all Docker instances on each node by running the following command:

```
#docker run -h <this-docker-instance-hostname> -it sequenceiq/hadoop-docker:2.7.0
/etc/bootstrap.sh -bash
```

You can start multiple Docker instances over the same physical node. This command starts a Docker instance with the hostname <this-docker-instance-hostname>.

9. For each Docker instance, change the `/etc/hosts` to map the Docker instance IP addresses to the hostname:

```
#vi /etc/hosts
172.17.0.2 node1docker1.gpfs.net node1docker1
172.17.0.4 node2docker1.gpfs.net node2docker1
172.17.0.6 node3docker1.gpfs.net node3docker1
```

**Note:** This must be done on the console of each Docker instance. You must add all Docker instances here if you want to set them up as one Hadoop cluster.

After a Docker instance is stopped, all changes are lost and you will have to make this change again after a new Docker instance has been started.

10. Select a Docker instance and start the Yarn ResourceManager on it:

```
#cd /usr/local/hadoop-2.7.0/sbin ./start-yarn.sh
```

You cannot run two ResourceManagers in the same Hadoop cluster. Therefore, you run this ResourceManager in the selected Docker instance.

11. Start Yarn NodeManager on other Docker instances by running the following command:

```
#/usr/local/hadoop-2.7.0/sbin/yarn-daemon.sh --config /usr/local/hadoop/etc/hadoop/
start nodemanager
```

12. Run `hadoop dfs -ls /` to check if you can run Map/Reduce jobs in Docker now. To stop the Yarn services running in Docker, perform the following steps:

```
->on Yarn ResourceManager Docker instance:
cd /usr/local/hadoop-2.7.0/sbin ./stop-yarn.sh
->on Yarn NodeManager Docker instances:
 /usr/local/hadoop-2.7.0/sbin/yarn-daemon.sh --config /usr/local/hadoop/etc/hadoop/
stop nodemanager
```

**Note:** While selecting HDFS transparency, the data locality is not supported for the Map/Reduce jobs running in Docker.

## The HDFS transparency federation

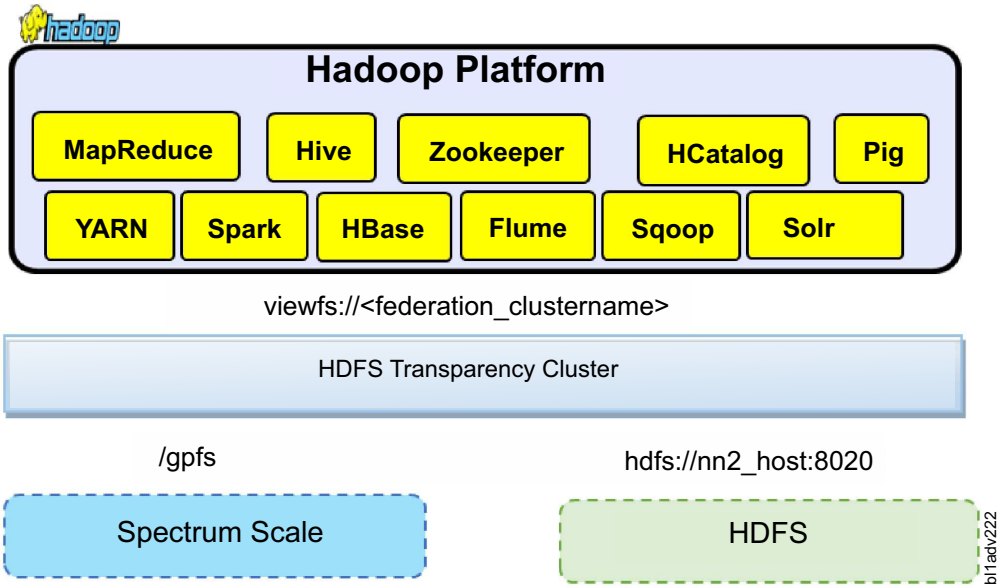
Federation has been introduced in HDFS to solve the HDFS NameNode scaling problem. This topic provides an overview of the HDFS Federation feature and the configuration and management of the federated cluster.

In HDFS transparency, federation is used to make the IBM Spectrum Scale file system coexist with the HDFS file system. For example, the Hadoop applications can get input from the native HDFS, analyze, and send the output to IBM Spectrum Scale. This feature is available in HDFS transparency 2.7.0-2 (`gpfs.hdfs-protocol-2.7.0-2`) and later. Also, the HDFS transparency federation can make two or more IBM Spectrum Scale file systems as one uniform file system for Hadoop applications. This is possible even if the file systems are from the same cluster as well as from different clusters. In a typical scenario, if you want to read data from an existing IBM Spectrum Scale file system, and analyze and send the analysis results to the new IBM Spectrum Scale file system.

### Federating IBM Spectrum Scale and HDFS

This topic lists the steps for federating IBM Spectrum Scale and HDFS.

Ensure that you have configured the HDFS Transparency cluster.




1. On one HDFS transparency cluster node, shut down the HDFS transparency cluster daemon by running the following command:

   **`mmhadoopctl connector stop`**

2. On *nn1_host*, add the following configuration settings in `/usr/lpp/mmfs/hadoop/etc/hadoop/core-site.xml`:

```
<configuration>
<property>
 <name>fs.defaultFS</name>
 <value>viewfs://<federation_clustername></value>
 <description>The name of the federation file system</description>
</property>

<property>
 <name>fs.viewfs.mounttable.<federation_clustername>.link./<mount_dir></name>
 <value>hdfs://nn1_host:8020/<mount_dir></value>
 <description>The name of the IBM Spectrum Scale file system</description>
</property>

<property>
 <name>fs.viewfs.mounttable.<federation_clustername>.link./<mount_dir></name>
 <value>hdfs://nn2_host:8020/<mount_dir></value>
 <description>The name of the hdfs file system</description>
</property>
</configuration>
```

**Note:** Change <federation_clustername> and <mount_dir> according to your cluster configuration: *nn1_host* and *nn2_host*. Here, *nn1_host* is the HDFS Transparency NameNode and *nn2_host* is the native HDFS NameNode.

3. On *nn1_host*, add the following configuration settings in /usr/lpp/mmfs/hadoop/etc/hadoop/hdfs-site.xml:

```
<configuration>
<property>
 <name>dfs.nameservices</name>
 <value>nn1,nn2</value>
</property>

<property>
 <name>dfs.namenode.rpc-address.nn1</name>
 <value>nn1-host:8020</value>
</property>

<property>
 <name>dfs.namenode.rpc-address.nn2</name>
 <value>nn2-host:8020</value>
</property>

<property>
 <name> dfs.namenode.http-address.nn1</name>
 <value>nn1-host:50070</value>
</property>

<property>
 <name>dfs.namenode.http-address.nn2</name>
 <value>nn2-host:50070</value>
</property>

</configuration>
```

4. On *nn1_host*, synchronize the configuration changes with the other HDFS transparency nodes by running the following command:

   **mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop/**

5. On *nn1_host*, start all the HDFS transparency cluster nodes by running the following command:

   **mmhadoopctl connector start**

6. Stop the Hadoop applications and the native HDFS services on the native HDFS cluster. The detailed command is dependent on the Hadoop distro. For example, for IBM BigInsights IOP, stop all the services on Ambari GUI.

7. Perform Step 2 and Step 3 on the node *nn2-host* with correct path of `core-site.xml` and `hdfs-site.xml` according to the Hadoop distro.

   If you take open source Apache Hadoop, the location of `core-site.xml` and `hdfs-site.xml` is `$YOUR_HADOOP_PREFIX/etc/hadoop/`. The $YOUR_HADOOP_PREFIX is the location of the Hadoop package. If you select IBM BigInsights IOP, you must update the configurations of `hdfs-site.xml` and `core-site.xml` on Ambari GUI.
8. Synchronize the updated configurations from the node *nn2-host* with all other native HDFS nodes and start the native HDFS services.

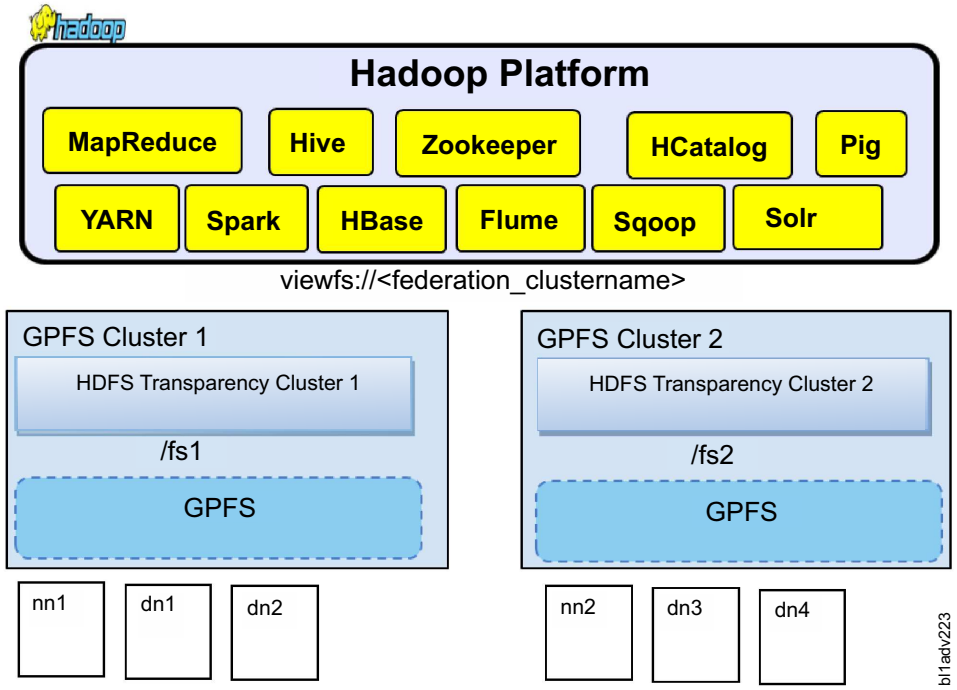
   If you select open source Apache Hadoop, you need to take scp to synchronize `core-site.xml` and `hdfs-site.xml` from the host *nn2-host* into all other native HDFS nodes. Then, start the native HDFS service by running `$YOUR_HADOOP_PREFIX/sbin/start-dfs.sh`.

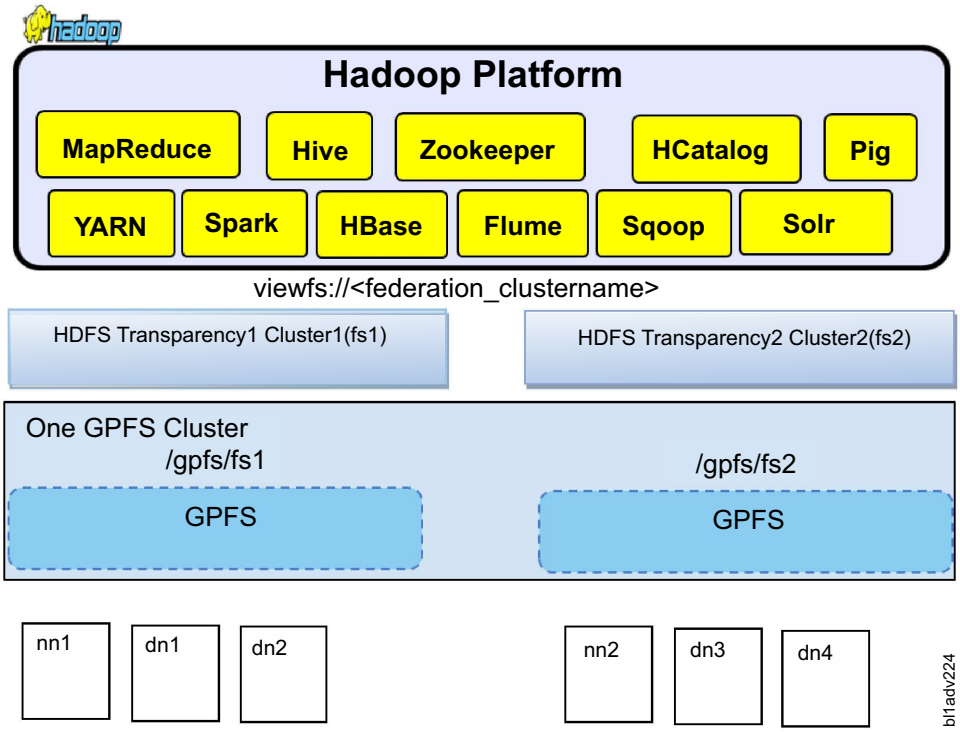
   If you take IBM BigInsights IOP, you need to restart the native HDFS on Ambari GUI. Ambari then synchronizes all configurations into all Hadoop nodes. If native HDFS services are up, you must shutdown native HDFS services again to ensure that all services on native HDFS are stopped. You could check the configurations under `/etc/hadoop/conf` to ensure that all changes have been synchronized into all nodes.
9. Update `core-site.xml` and `hdfs-site.xml` for the Hadoop clients on which the hadoop applications will run over federation. If you select open source Apache Hadoop, the location of `core-site.xml` and `hdfs-site.xml` is `$YOUR_HADOOP_PREFIX/etc/hadoop/`. The $YOUR_HADOOP_PREFIX is the location of the Hadoop package. If you take other Hadoop distro, see "Known limitations" on page 513.
10. To ensure that the federated fs is functioning correctly, run the **`hadoop fs –ls /`** command.

## Spectrum Scale file systems federation

You can federate two IBM Spectrum Scale file systems from different clusters or from the same cluster.

Irrespective of the mode that you select, configure one HDFS transparency cluster for each IBM Spectrum Scale file system, and then federate the two HDFS transparency clusters together.






To federate two file systems from the same cluster, select nodes that can provide HDFS transparency services for the first file system and the second file system separately.

**Configuring the federation:**

This topic describes the steps to configure the federation

Before configuring the federation, configure HDFS transparency cluster 1 and HDFS transparency cluster 2 for each file system.

1. To stop the services, run the **mmhadoopctl** connector stop on both HDFS transparency clusters.
2. On the *nn1* host, add the following configuration settings in `/usr/lpp/mmfs/hadoop/etc/hadoop/core-site.xml`:

```
<configuration>
<property>
 <name>fs.defaultFS</name>
 <value>viewfs://<federation_clustername></value>
 <description>The name of the federation file system</description>
</property>

<property>
 <name>fs.viewfs.mounttable.<federation_clustername>.link./<mount_dir></name>
 <value>hdfs://nn1_host:8020/<mount_dir></value>
 <description>The name of the gpfs file system</description>
</property>


<property>
 <name>fs.viewfs.mounttable.<federation_clustername>.link./<mount_dir></name>
 <value>hdfs://nn2_host:8020/<mount_dir></value>
 <description>The name of the hdfs file system</description>
</property>
</configuration>
```

**Note:** Change <federation_clustername> and <mount_dir> according to your cluster. Change *nn1_host* and *nn2_host* accordingly.

3. On *nn1_host*, add the following configuration settings in hdfs-site.xml.

```
<configuration>
<property>
 <name>dfs.nameservices</name>
 <value>nn1,nn2</value>
</property>

<property>
 <name>dfs.namenode.rpc-address.nn1</name>
 <value>nn1-host:8020</value>
</property>

<property>
 <name>dfs.namenode.rpc-address.nn2</name>
 <value>nn2-host:8020</value>
</property>

<property>
 <name> dfs.namenode.http-address.nn1</name>
 <value>nn1-host:50070</value>
</property>

<property>
 <name>dfs.namenode.http-address.nn2</name>
 <value>nn2-host:50070</value>
</property>
</configuration>
```

4. On *nn1_host*, synchronize the configuration change to another HDFS transparency cluster node:

   **mmhadoopctl connector syncconf /usr/lpp/mmfs/hadoop/etc/hadoop/**

5. On *nn2_host*, perform Step 1 through Step 4.
6. On *nn1_host*, start the HDFS transparency cluster by running the following command:

   **mmhadoopctl connector start**

7. On *nn2_host*, start the other HDFS transparency cluster by running the following command:

   **mmhadoopctl connector start**

8. Update core-site.xml and hdfs-site.xml for the Hadoop clients on which the Hadoop applications run over federation.

   If you take open source Apache Hadoop, the location of core-site.xml and hdfs-site.xml is $YOUR_HADOOP_PREFIX/etc/hadoop/. The $YOUR_HADOOP_PREFIX is the location of the Hadoop package. If you take another Hadoop distro, see "Known limitations."

9. Restart the Hadoop applications on both clusters.

   **Note:** If you select HDFS Transparency, you must always keep the native HDFS non-functional.

10. To ensure that the federated fs is functioning correctly, run the **hadoop fs –ls /** command.

## Known limitations

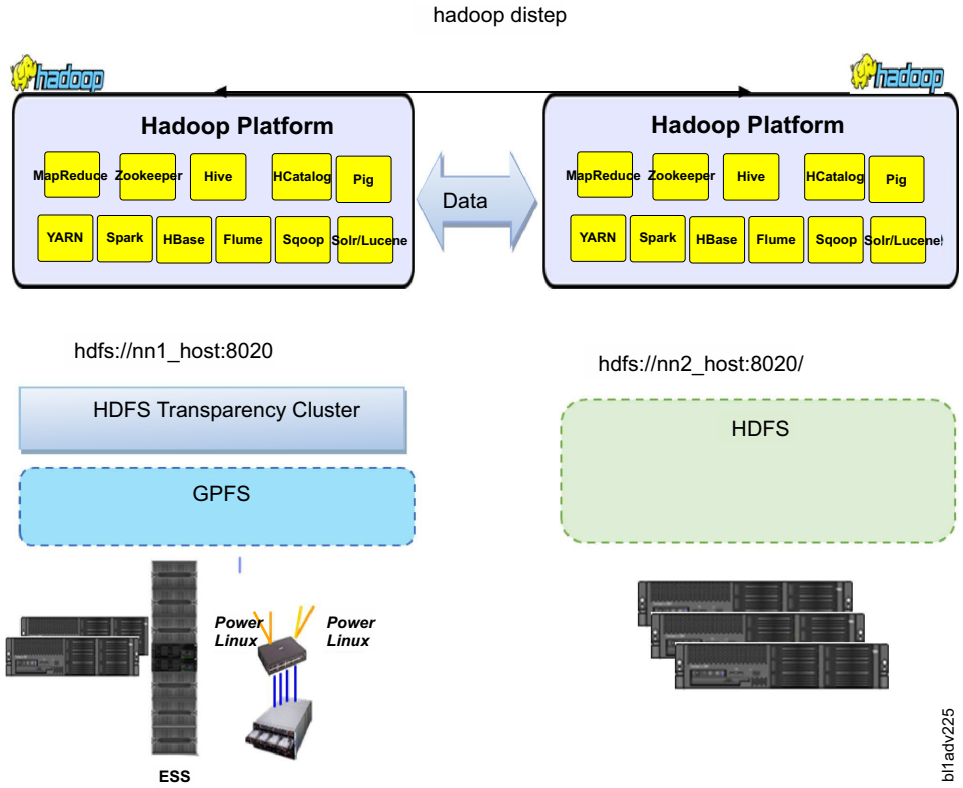
- All the changes in /usr/lpp/mmfs/hadoop/etc/hadoop/core-site.xml and /usr/lpp/mmfs/hadoop/etc/hadoop/hdfs-site.xml must be updated in the configuration file that is used by the Hadoop distro. However, Hadoop distro occasionally manages the configuration, and the management interface might not support the key used for federation. IBM BigInsights IOP takes Ambari and Ambari GUI does not support some property names (see Ambari-15455).

  If you want to set up the federation for IBM BigInsights IOP, send an email to scale@us.ibm.com.
- The native HDFS and HDFS transparency cannot be run over the same node because of the network port number conflict.

- Before being federated to GPFS, native HDFS must be in the running state. Otherwise, HDFS reports an exception.
- Start and stop the native HDFS cluster or the HDFS Transparency cluster separately if you want to maintain them.

## Hadoop distcp support

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




There are no additional configuration changes. The **hadoop distcp** command is supported in HDFS transparency 2.7.0-2 (gpfs.hdfs-protocol-2.7.0-2) and later.

```
hadoop distcp hdfs://nn1_host:8020/source/dir hdfs://nn2_host.:8020/target/dir
```

### Known Issues and Workaround

### Issue 1: Permission is denied when the hadoop distcp command is run with the root credentials.

The super user root in Linux is not the super user for Hadoop. If you do not add the super user account to gpfs.supergroup, the system displays the following error message:

```
org.apache.hadoop.security.AccessControlException: Permission denied: user=root, access=WRITE,
inode="/user/root/.staging":hdfs:hdfs:drwxr-xr-x
```

at

```
org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(FSPermissionChecker.java:319).
```

**Workaround**

Add the super user account to gpfs.supergroup in gpfs-site.xml to configure the root as the super user or run the related **hadoop distcp** command with the super user credentials.

### Issue 2: Access time exception while copying files from IBM Spectrum Scale to HDFS with the -p option

```
[hdfs@c8f2n03 conf]$ hadoop distcp -overwrite -p hdfs://c16f1n03.gpfs.net:8020/testc16f1n03/
hdfs://c8f2n03.gpfs.net:8020/testc8f2n03
```

Error: `org.apache.hadoop.ipc.RemoteException(java.io.IOException)`: Access time for HDFS is not configured. Set the **dfs.namenode.accesstime.precision** configuration parameter at `org.apache.hadoop.hdfs.server.namenode.FSDirAttrOp.setTimes(FSDirAttrOp.java:101)`

**Workaround**

Change the `dfs.namenode.accesstime.precision` value from 0 to a value such as 3600000 (1 hour) in `hdfs-site.xml` for the HDFS cluster.

### Issue 3: The distcp command fails when the src director is root.

```
[hdfs@c16f1n03 root]$ hadoop distcp hdfs://c16f1n03.gpfs.net:8020/ hdfs://c8f2n03.gpfs.net:8020/test5
16/03/03 22:27:34 ERROR tools.DistCp: Exception encountered
java.lang.NullPointerException

at org.apache.hadoop.tools.util.DistCpUtils.getRelativePath(DistCpUtils.java:144)

at org.apache.hadoop.tools.SimpleCopyListing.writeToFileListing(SimpleCopyListing.java:353)
```

**Workaround**

Specify at least one directory or file at the source directory.

### Issue 4: The distcp command throws NullPointerException when the target directory is root in the federation configuration but the job is completed

The **hadoop distcp** command throws NullPointerException when the target directory in the federation configuration is root, and the job is completed. For more details, see https://issues.apache.org/jira/browse/HADOOP-11724.

## Automatic Configuration Refresh

The Automatic configuration refresh feature is supported in gpfs.hdfs-protocol 2.7.0-2 and later.

After making configuration changes in `/usr/lpp/mmfs/hadoop/etc/hadoop` or in the IBM Spectrum Scale file system, such as maximum number of replica and NSD server, run the following command to refresh HDFS transparency without restarting the HDFS transparency services:

```
/usr/lpp/mmfs/hadoop/bin/gpfs dfsadmin -refresh <namenode_hostname>:<port> refreshGPFSConfig
```

Run the command on an HDFS transparency node and change *<namenode_hostname>:<port>* according to the HDFS transparency configuration. For example, if *fs.defaultFS* is `hdfs://c8f2n03.gpfs.net:8020` in `/usr/lpp/mmfs/hadoop/etc/hadoop/core-site.xml`, replace *<namenode_hostname>* with `c8f2n03.gpfs.net` and *<port>* with `8020`. HDFS transparency synchronizes the configuration changes with the HDFS transparency services running on the HDFS transparency nodes.

## Application interaction with HDFS transparency

The Hadoop application interacts with the HDFS transparency similar to their interactions with native HDFS. They can access data in the IBM Spectrum Scale filesystem using Hadoop file system APIs and Distributed File System APIs.

The application might have its own cluster that is larger than HDFS transparency cluster. However, all the nodes within the application cluster must be able to connect to all nodes in HDFS transparency cluster by RPC.

Yarn can define the nodes in cluster by using the slave files. However, HDFS transparency can use a set of configuration files that are different from yarn. In that case, slave files in HDFS transparency can be different from the one in the yarn.

### Application interface of HDFS transparency

In HDFS transparency, applications can use the APIs defined in org.apache.hadoop.fs.FileSystem class and org.apache.hadoop.fs.AbstractFileSystem class to access the file system.

### Command line for HDFS transparency

You can use the HDFS shell command line with the HDFS transparency.

You can access commands from the HDFS command shell:

```
$YOUR_HADOOP_PREFIX/bin/hdfs
Usage: hdfs [--config confdir] COMMAND
       where COMMAND is one of:
dfs                   run a filesystem command on the file systems supported in Hadoop.
namenode -format      format the DFS filesystem
secondarynamenode     run the DFS secondary namenode
namenode              run the DFS namenode
journalnode           run the DFS journalnode
zkfc                  run the ZK Failover Controller daemon
datanode              run a DFS datanode
dfsadmin              run a DFS admin client
haadmin               run a DFS HA admin client
fsck                  run a DFS filesystem checking utility
balancer              run a cluster balancing utility
jmxget                get JMX exported values from NameNode or DataNode.
mover                 run a utility to move block replicas across
                      storage types
oiv                   apply the offline fsimage viewer to an fsimage
oiv_legacy            apply the offline fsimage viewer to an legacy
                      fsimage
oev                  apply the offline edits viewer to an edits file
fetchdt              fetch a delegation token from the NameNode
getconf              get config values from configuration
groups               get the groups which users belong to
snapshotDiff         diff two snapshots of a directory or diff the
                     current directory contents with a snapshot
lsSnapshottableDir   list all snapshottable dirs owned by the current user
                     Use -help to see options
portmap              run a portmap service
nfs3                 run an NFS version 3 gateway
cacheadmin           configure the HDFS cache
crypto               configure HDFS encryption zones
storagepolicies      list/get/set block storage policies
version              print the version
```

Most commands print help when invoked without parameters.

**Note:** All commands from **hdfs dfs** are supported (`hdfs dfs -du` and `hdfs dfs -df` are not exact in the output, use **du** or **df/mmdf** for exact output). Other commands from HDFS interface are not supported (For example, `hdfs namenode –format`) because these commands are not needed for IBM Spectrum Scale.

## Security

HDFS transparency supports full Kerberos and it is verified over IBM BigInsights IOP 4.1.

Refer to the IBM Spectrum Scale HDFS transparency Security Guide.

### Hadoop Data Access Audit

HDFS Transparency is certificated with IBM Security Guardium® DAM (Database Activity Monitoring) to monitor the Hadoop Data Access over IBM Spectrum Scale. For more information, see Big data security and auditing with IBM InfoSphere Guardium.

## Removing and upgrading HDFS transparency cluster

Before upgrading the HDFS transparency, you need to remove the older IBM Spectrum Scale Hadoop connector.

### Removing IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.4, 4.1.0.5, 4.1.0.6, or 4.1.0.7 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.4, 4.1.0.5, 4.1.0.6, or 4.1.0.7 releases, this section explains the steps required to remove the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

1. Remove any links or copies of the `hadoop-gpfs-2.4.jar` file from your Hadoop distribution directory. Also, remove any links or copies of the `libgpfshadoop.64.so` file from your Hadoop distribution directory.

   **Note:** For IBM BigInsights IOP 4.0, the distribution directory is `/usr/iop/4.0.0.0`.
2. Stop the current connector daemon:

   ```
   # ps -elf | grep gpfs-connector-daemon
   # kill -9 <pid-of-connector-daemon>
   ```
3. Run the following commands, to remove callbacks from IBM Spectrum Scale:

   ```
   cd /usr/lpp/mmfs/fpo/hadoop-2.4/install_script
   ./gpfs-callbacks.sh --delete
   ```

   Run the **`mmlscallbacks all`** command to check whether connector-related callbacks, such as `callback ID start-connector-daemon` and `stop-connector-daemon`, are removed. The IBM Spectrum Scale Hadoop connector callbacks are cluster-wide and this step is required to be done over any one of nodes.
4. Remove the following files:

   ```
   rm -f /var/mmfs/etc/gpfs-callbacks.sh
   rm -f /var/mmfs/etc/gpfs-callback_start_connector_daemon.sh
   rm -f /var/mmfs/etc/gpfs-callback_stop_connector_daemon.sh
   rm -f /var/mmfs/etc/gpfs-connector-daemon
   ```
5. Remove the IBM Spectrum™ Scale-specific configuration from your Hadoop `core-site.xml` file. Modify the **`fs.defaultFS`** into an HDFS schema format after removing the following configurations:

   ```
   fs.AbstractFileSystem.gpfs.impl, fs.AbstractFileSystem.hdfs.impl, fs.gpfs.impl, fs.hdfs.impl,
   gpfs.mount.dir, gpfs.supergroup
   ```

Install and setup the HDFS transparency, see the “Upgrading HDFS transparency cluster” on page 518 for more information.

### Removing IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 over IBM Spectrum Scale 4.1.0.7 or 4.1.0.8 releases, this section explains the steps required to remove the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. **rpm -e gpfs.hadoop-2-connector**
4. Remove the IBM Spectrum Scale-specific configuration from your Hadoop `core-site.xml` file. Modify the **fs.defaultFS** into an HDFS schema format by removing the following configurations:

   ```
   fs.AbstractFileSystem.gpfs.impl, fs.AbstractFileSystem.hdfs.impl, fs.gpfs.impl, fs.hdfs.impl,
   gpfs.mount.dir, gpfs.supergroup
   ```

Install and setup the HDFS transparency, see the "Upgrading HDFS transparency cluster" for more information.

## Removing IBM Spectrum Scale Hadoop connector 2.4 or 2.5 over IBM Spectrum Scale 4.1.1 and later releases

For users who are using IBM Spectrum Scale Hadoop connector 2.4 or 2.5 in IBM Spectrum Scale 4.1.1 and later, this section explains the steps required to remove the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. **rpm -e gpfs.hadoop-2-connector**
4. Remove the IBM Spectrum Scale-specific configuration from your Hadoop `core-site.xml` file. Modify the **fs.defaultFS** into an HDFS schema format by removing the following configurations:

   ```
   fs.AbstractFileSystem.gpfs.impl, fs.AbstractFileSystem.hdfs.impl, fs.gpfs.impl, fs.hdfs.impl,
   gpfs.mount.dir, gpfs.supergroup
   ```

Install and setup the HDFS transparency, see the "Upgrading HDFS transparency cluster" for more information.

## Removing IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7, 4.1.0.8, or 4.1.1 and later releases

For users who are using IBM Spectrum Scale Hadoop connector 2.7 (earlier release) over IBM Spectrum Scale 4.1.0.7, 4.1.0.8, or 4.1.1 and later releases, this section explains the steps required to remove the old connector over each node in the cluster.

If you are using Hadoop 1.x, IBM Spectrum Scale Hadoop Connector 2.7 does not support Hadoop 1.x and you need to upgrade your Hadoop version first.

1. **mmhadoopctl connector stop**
2. **mmhadoopctl connector detach --distribution BigInsights**
3. **rpm -e gpfs.hadoop-2-connector**
4. Remove the IBM Spectrum Scale-specific configuration from your Hadoop `core-site.xml` file. Modify the **fs.defaultFS** into an HDFS schema format by removing the following configurations:

   ```
   fs.AbstractFileSystem.gpfs.impl, fs.AbstractFileSystem.hdfs.impl, fs.gpfs.impl, fs.hdfs.impl,
   gpfs.mount.dir, gpfs.supergroup
   ```

For more information on installing and setting up the HDFS transparency, see the "Upgrading HDFS transparency cluster."

## Upgrading HDFS transparency cluster

This section explains how to upgrade HDFS transparency cluster.

1. Back up the configuration, in case of any failures.

2. Stop the HDFS transparency service on all nodes using the command: **/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector stop**
3. Upgrade the RPM on each node by using the command: **rpm –U gpfs.hdfs-protocol-2.7.0-<x>.x86_64.rpm**. It does not update any configuration files under the /usr/lpp/mmfs/hadoop/etc/hadoop folder.

   The core-site.xml, hdfs-site.xml, and slaves files are not removed during the upgrade.
4. Start HDFS transparency service on all nodes using the command: **/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl connector start**

## Limitation and difference from HDFS

This section describes the limitations and the configurations in IBM Spectrum Scale that are different from HDFS.

| Property name | Value | Limitation |
|---|---|---|
| dfs.permissions.enabled | True/false | For HDFS transparency, permission check is always done. |
| dfs.namenode.acls.enabled | True/false | In native HDFS, the NameNode manages all metadata including ACL information. So, HDFS can use this to turn on or off the ACL checking. However, for IBM Spectrum Scale, HDFS transparency does not hold the metadata. When it is on, the ACL is set and stored in the IBM Spectrum Scale file system. If the administrator turns this off later, the ACL entries set before are still stored and active in IBM Spectrum Scale. |
| dfs.blocksize | Long digital | Must be an integer multiple of the IBM Spectrum Scale filesystem block size, **mmlsfs -B**. The maximum value is 1024 * file-system-data-block-size. |
| Spectrum Scale.data.dir | String | All users of Hadoop must have full access to this directory. If this configuration is omitted, then users of Hadoop must have full access to the Spectrum Scale.mount.dir directory. |
| dfs.namenode.fs-limits.max-xattrs-per-inode | INT | Does not apply to the HDFS transparency. |
| dfs.namenode.fs-limits.max-xattr-size | INT | Does not apply to the HDFS transparency. |

### Functional limitations

1. Maximum number of extended attributes is limited by IBM Spectrum Scale. The total size of the extended attribute key and value should be less than a metadata block size in IBM Spectrum Scale.
2. Extended attributes operation on snapshots is not supported now.
3. Raw namespace is not implemented as it is not used internally.

# HDFS transparency security

## Configuration and binary permissions

All configuration files for HDFS transparency are located in the /usr/lpp/mmfs/hadoop/etc/hadoop folder after installation. Configuration files can be read and modified only by the root user.

**Note:** For security considerations, the root user must not grant read and write permissions to the non-root users.

The following example shows the output of the **ls -la** command:

```
/usr/lpp/mmfs/hadoop]# ls –la
drwx------  3 root root 4096 Nov  9 09:56 etc
```

The output of the **ls -la** command displays the permissions of the HDFS transparency scripts:

```
/usr/lpp/mmfs/hadoop/bin]# ls –la
-r-xrxr-x 1 root root 4484 Nov  6 10:38 gpfs

/usr/lpp/mmfs/hadoop/sbin
[root@c8f2n09 sbin]# ls –la
total 48
drwxr-xr-x  2 root root 4096 Nov 16 05:21 .
drwxr-xr-x 10 root root 4096 Nov 16 05:38 ..
-r-x------  1 root root 3310 Nov 16 05:20 deploy-gpfs.sh
-r-xr-xr-x  1 root root  697 Nov 16 05:20 gpfs-state.sh
-r-xr-xr-x  1 root root 5380 Nov 16 05:20 hadoop-daemon.sh
-r-xr-xr-x  1 root root 1360 Nov 16 05:20 hadoop-daemons.sh
-r-xr-xr-x  1 root root 4959 Nov 16 05:20 mmhadoopctl
-r-xr-xr-x  1 root root 2145 Nov 16 05:20 slaves.sh
-r-x------  1 root root 1111 Nov 16 05:20 start-gpfs.sh
-r-x------  1 root root  740 Nov 16 05:20 stop-gpfs.sh
```

The root user must keep the permissions of all the configuration files unchanged after the installation.

**Note:** The root user must not grant the write permission to the non-root users.

The root user must start the connector because the Java binaries check the UID of the user that starts the connector and exits when the UID does not belong to a root user. Users other than root user cannot start or stop the HDFS transparency service because the HDFS transparency binary code checks the UID of the user. If the user who starts the service is not a root user, it exits.

The non-root users can run the **mmhadoopctl connector getstate** command to view the state of the connector. The read and execute permissions of the gpfs-state.sh, hadoop-daemon.sh, hadoop-daemons.sh, and slaves.sh files can be used by the non-root users to view the state of the connector.

**Note:** By default, HDFS transparency installs the above scripts with the default permissions. To avoid security vulnerability, the cluster administrators must ensure that the permissions for these files are not changed.

## HDFS transparency daemon UID/GID and Hadoop super groups

HDFS transparency has two types of daemons: NameNode and DataNode. Both of these daemons can only be started by the root user because certain file operations, such as setPermission and setOwner, in the Hadoop distributed file system API need root privileges.

HDFS transparency binaries exit immediately when the login credentials do not match the UID/GID of the root user. The **dfs.permissions.superusergroup** parameter in hdfs-site.xml and the **gpfs.supergroup** parameter in /usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml are used by the customers to configure the Hadoop super group.

The **dfs.permissions.superusergroup** parameter can be configured as a single group and **gpfs.supergroup** can be a comma-separated group list. All users in Hadoop super groups have super privileges in the Hadoop cluster, like the super user root in Linux/Unix OS.

For non-Hadoop super users, when HDFS transparency receives RPC requests from the HDFS clients, HDFS transparency creates new threads called setfsuid/setfsgid. HDFS transparency creates these threads to replace the user ID or the group ID of the threads with the user ID or group ID of the client and handle the requests. This can restrict the privileges of a common user in the Hadoop cluster.

For Hadoop super users, all operations are performed under the security context of the root user. The user must configure Hadoop super groups carefully.

## The simple security mode

When Kerberos is not enabled, Hadoop runs under the simple security mode. In this mode, RPCs are not encrypted and authenticated, and all users can submit maps and reduce jobs to the Hadoop cluster. A Hadoop cluster running in the simple security mode is vulnerable to attack via the network from outside the clusters and from users logged on to the nodes in the cluster.

**Note:** You must enable Kerberos. For more information about Kerberos, see "The Kerberos mode" on page 522. The data transfers and RPCs from the clients to the NameNode and DataNode are not encrypted, and therefore, vulnerable to attack through the network.

In the Hadoop cluster, the user ID must be created on all the nodes including the nodes submitting jobs and the nodes running NameNode and DataNode. If a user submits the jobs, but the user ID is not created in DataNodeX, Map and Reduce jobs on DataNodeX cannot access the data because Hadoop creates the files with the user ID and group ID of the user. Hadoop itself does not manage user IDs and group IDs. Hadoop only transfers the user ID and group ID from the job submitter and stores them as the owner of the job output file in the file system. For a user to read or write a file, permissions need to be set using the traditional Linux/Unix permission control. The authentication of a user is done using Linux authentication. If a user has successfully logged on to the system, the user has passed the OS authentication,

The **fs.permissions.umask-mode** parameter in hdfs-site.xml can be configured as the umask used while creating files and directories. For more information about this configuration, see Hadoop website.

For more information about security configurations, see HDFS Permission Guide and HDFS Permissions and Security Guide.

**ACL:**

The **dfs.namenode.acls.enabled** property in hdfs-site.xml can be used to enable support for ACLs by HDFS transparency.

**Note:** Hadoop only supports POSIX ACL. If the applications set NFS ACL for certain files through the POSIX interface, jobs fail while handling the ACL of those files and java exceptions are reported in the GPFS HDFS transparency logs.

**Namenode block access token:**

In the previous releases of Hadoop, DataNode did not enforce access control on the data blocks. If an unauthorized client provided the block ID, the client could read a data block. Also, unauthorized users were able to write data blocks to DataNodes.

In Hadoop Release 0.2x and later, for HDFS transparency, when clients request to access files, the file permissions are checked. Only if the client has the required permissions, NameNode returns a token in

the HMAC-SHA1 format to the client. The client sends the token back to DataNode when it requests data access. DataNode checks this token and grants or refuses access to the block.

To enable the NameNode block access token, configure the following settings in the hdfs-site.xml file:

```
dfs.block.access.token.enable=yes
dfs.block.access.key.update.interval=600 (by default, minutes)
dfs.block.access.token.lifetime=600 (by default, minutes)
```

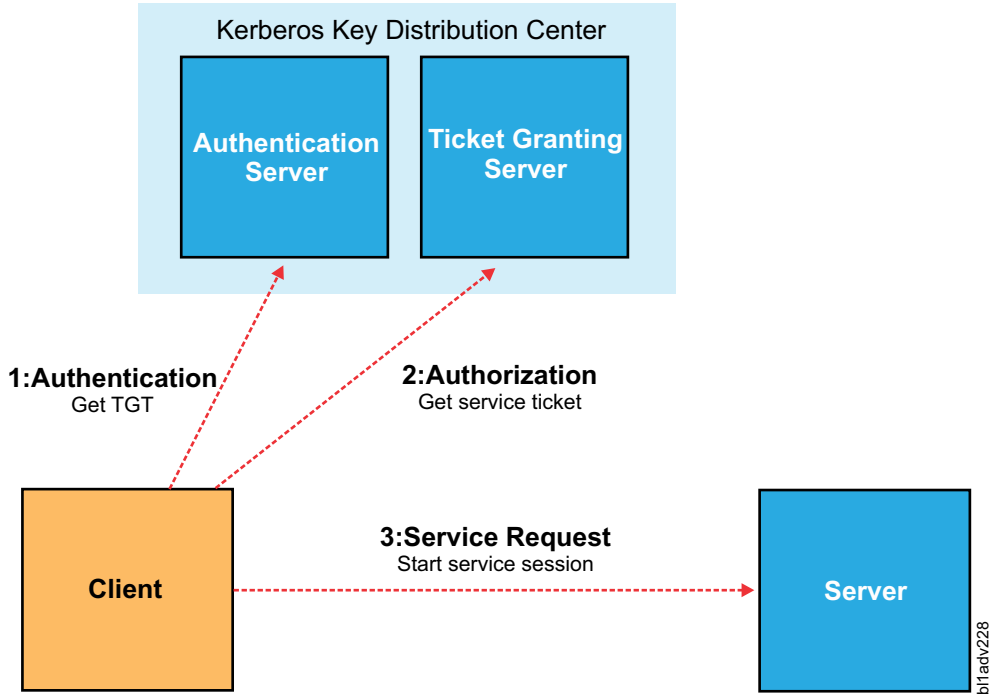
**Note:** By default, this feature is enabled in the IBM BigInsight IOP distribution. However, this feature cannot prevent the attacker from connecting to NameNode if Kerberos is not enabled.

## 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.

The Kerberos service comprises of a client-server architecture that provides secure transactions over networks. The service offers strong user authentication, as well as integrity and privacy. The authentication verifies the identities of the sender and the receiver in a network transaction. The service also checks for data integrity and encrypts the data during transmission.

Using the Kerberos service, you can log on to other machines, execute commands, exchange data, and transfer files securely. Additionally, Kerberos provides authorization services that allow administrators to restrict access to services and machines.




So, in the Kerberos mode, only authorized users can access services, thereby preventing an unauthorized access to services. The Kerberos mode also encrypts the data during transmission to avoid data exposure.

To enable Kerberos, configure the core-site.xml as follows:

```
hadoop.security.authorization=true
hadoop.security.authentication=Kerberos (the default is "simple")
```

**SASL/GSS API and RPC:**

This topic describes the server-side authentication, client-side authentication, and Hadoop client, Hadoop server, and RPC

**Server-side authentication**

Hadoop services, such as NameNode and DataNode, in HDFS transparency must authenticate to Kerberos KDC. During the startup, the service will log in KDC by using the service principal and the keytab configured in the core-site.xml file.

**Note:** All keytab files used by Hadoop services are stored in a local file system of the node running the services. Different Hadoop distro might take different locations for this. For IBM BigInsights IOP, the distro will take `/etc/security/keytabs/` on the nodes that are running the services. Different keytab files are owned by different users and are readable only by the owner of the file. Hadoop cluster administrator must be careful and must not expose the read and write permission of these files to other users.

After the service authentication check passes, the service finishes the start-up procedure and is ready to handle the client requests.

**Client-side authentication**

A Hadoop user must be authenticated by the Kerberos KDC before accessing the Hadoop services through the client tool by using their own user principal.

The steps for Hadoop client to submit jobs:

1. Log on to a client machine that is connected to the Hadoop cluster, and then execute the **`kinit`** command with the principal and the password.
2. The **`kinit`** command authenticates the user with the KDC, gets the Kerberos TGT ticket, and puts the ticket into the ticket cache in the local file system.
3. Run the client tools. For example, submit a MapReduce job through the JobClient.

The client bootstraps and issues connection requests to the server side.

**Hadoop client, Hadoop server, and RPC**

After the server and client sides authenticate with Kerberos successfully, the server waits for the client requests. After the client issues a request, both server and client come down to the SASL/GSSAPI stack:

1. The client stack picks up the client TGT ticket in the current access control context.
2. Using the TGT, the client requests a service ticket from the KDC targeting the right service or server that the user or the client software is accessing.
3. The client sends out the service ticket first as part of the connector with the service. The server or service decrypts the service ticket with the service key. The service provides the service key when it authenticates with KDC. If the server can decrypt the service ticket successfully, it means that the client has passed the authentication.

The workflow in the SASL/GSSAPI stack regarding SASL and GSSAPI specifications involving Kerberos is complex, but it is not just for authentication, as it also builds a secure context and channel for both the server and client sides.

Three levels of protection are provided by this stack: auth (authentication), int (integrity), and privacy (encryption). These options are exposed and can be configured in the latest version of Hadoop making the encryption of the RPC channel easy.

This feature is controlled by **hadoop.rpc.protection** in core-site.xml. The authentication value is for enabling SASL connections for authentication, the integrity value is for enabling SASL connections for authentication and data integrity, and the privacy value is for enabling SASL connections for authentication, data integrity, and privacy (encrypting the data).

**Delegated NameNode token:**

All operations from client nodes must be connected with the KDC to be authenticated when Kerberos is enabled. This process can impact the performance of the Hadoop jobs. Therefore, the NameNode Token delegation was introduced to reduce the performance impact from Kerberos and the load on the KDC server.

Authenticating clients through delegated NameNode tokens is a two-way authentication protocol that is based on Java SASL Digest-MD5. The token is obtained during job submissions, and then submitted to JobTracker. The steps are as follows:

1. The user authenticates the JobTracker by using Kerberos.
2. By using Kerberos, the user authenticates the NameNode(s) that the tasks will interact with at runtime. The user gets a delegation token from each of the NameNodes.
3. The user passes the tokens to the JobTracker as part of the job submission.

All TaskTrackers running the job tasks get a copy of the tokens through an HDFS location that is private to the user that the MapReduce daemons run. The tokens are written to a file in a private area that is visible to the job-owner user on the TaskTracker machine.

While launching the task, the TaskTracker exports the location of the token file as an environment variable. The task process loads the tokens into the memory. The file is read as part of the static initialization of the `UserGroupInformation` class used in the Hadoop services. This information is useful for the RPC client.

In the Kerberos mode, the Apache Hadoop RPC client can communicate securely with a server by using either tokens or Kerberos. The RPC client is programmed in a way that when a token exists for a service, it will be used for secure communication. If no token is available, Kerberos is used.

When Kerberos is enabled, Delegated NameNode token takes effect automatically. The configurations settings related to this feature are in the hdfs-site.xml file:

```
dfs.namenode.delegation.key.update-interval(milliseconds)
dfs.namenode.delegation.token.max-lifetime(milliseconds)
dfs.namenode.delegation.token.renew-interval(milliseconds)
```

**HTTP SPNEGO authentication:**

By default, Hadoop web applications such as ResourceManager, NodeNodeManager, JobTracker, NameNode, TaskTrackers, and DataNodes can be accessed without authentication. If Kerberos is enabled, all web applications can be configured to authenticate through Kerberos HTTP SPNEGO.

The configurations settings for this feature can be viewed in core-site.xml and the following properties values must be changed:

```
<!--  HTTP web-consoles Authentication -->
  <property>
<name>hadoop.http.filter.initializers</name>

<value>org.apache.hadoop.security.AuthenticationFilterInitializer</value>
  </property>

  <property>
  <name>hadoop.http.authentication.type</name>
  <value>kerberos</value>
```

```
</property>

<property>
<name>hadoop.http.authentication.token.validity</name>
<value>36000</value>
</property>

<property>
<name>hadoop.http.authentication.signature.secret.file</name>
<value>/hadoop/hadoop/conf/http-secret-file</value>
</property>

<property>
<name>hadoop.http.authentication.cookie.domain</name>
<value></value>
</property>

<property>
<name>hadoop.http.authentication.simple.anonymous.allowed</name>
<value>false</value>
</property>

<property>
<name>hadoop.http.authentication.kerberos.principal</name>
<value>HTTP/hz169-91.i.site.com@I.NETEASE.COM</value>
</property>

<property>
<name>hadoop.http.authentication.kerberos.keytab</name>
<value>/hadoop/hadoop/conf/http.keytab</value>
</property>
```

**RPC and data encryption:**

To encrypt data that is transferred between Hadoop services and clients, set **hadoop.rpc.protection** to privacy in core-site.xml.

To activate data encryption for the data transfer protocol of DataNode, set **dfs.encrypt.data.transfer** to true in hdfs-site.xml. Optionally, set **dfs.encrypt.data.transfer.algorithm** to either 3DES or RC4 to choose the specific encryption algorithm. If the encryption algorithm is not specified then the default value configured for JCE, which is usually 3DES, is used for the system. Setting **dfs.encrypt.data.transfer.cipher.suites** to AES/CTR/NoPadding activates AES encryption. By default, this is not specified. Therefore, AES is not used. When AES is specified, the algorithm specified in **dfs.encrypt.data.transfer.algorithm** is still used during the initial key exchange. The AES key bit length can be configured by setting **dfs.encrypt.data.transfer.cipher.key.bitlength** to 128, 192, or 256. The default value is 128.

AES offers the greatest cryptographic strength and the best performance. At this time, 3DES and RC4 are most commonly used in Hadoop clusters.

Data transfers between Web console and clients are protected using SSL (HTTPS), such as httpfs and webHDFS.

If your Hadoop cluster has to be NIST-compliant, you must select NIST-compliant encryption algorithm and key length. For NIST-compliant algorithm and key length, see Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths. Also, you must configure the IBM Spectrum Scale cluster as NIST-compliant by running the **mmchconfig nistCompliance=SP800-131A** command in the IBM Spectrum Scale cluster.

**Note:** 3DES and AES are NIST-compliant whereas RC4 is not NIST-compliant.

## Shortcircuit and security

In HDFS, shortcircuit can be enabled when a client and DataNode are on the same node. By enabling shortcircuit, an application that needs to read a file can obtain the file descriptor from the DataNode and read the data block directly. Shortcircuit reads provide a significant boost in the read I/O performance. The Hadoop client can only read data from the file descriptor because the DataNode opens the file in the read-only mode.

For HDFS transparency with FPO configuration, this feature can be enabled for enhanced performance. In the shortcircuit mode, the DataNode and the client communicate through a Unix domain socket that is configured through `dfs.domain.socket.path` in hdfs-site.xml, configured as `/var/lib/hadoop-hdfs/dn_socket`.

```
/var/lib/hadoop-hdfs]# ls -l
drwxr-xr-x  2 root root  4096 Nov  5 01:04 hadoop-hdfs
srw-rw-rw-  1 root root     0 Nov  5 01:04 dn_socket
```

The permission for the socket file must be 666 so that all common users can read the socket and receive messages from the DataNode. The x permission bit does not matter here

When Kerberos is enabled, the Kerberos server authenticates the Hadoop client and the DataNode checks the authorization of the Hadoop client by checking the service ticket. Whether Kerberos is enabled or not, the DataNode checks the Block Access Token from the Hadoop client and ensures that the Hadoop client can access the target file before the file descriptor is sent to the Hadoop client. These checks ensure that the file descriptor is not sent to invalid users.

**Note:** The **`dfs.block.access.token.enable`** parameter must be configured as true when shortcircuit is enabled.

Also, the message transfer over the Unix domain socket in shortcircuit is not encrypted. For more information about security considerations in shortcircuit, see HDFS 5353. However, as the data is transferred over the same machine and not over the TCP/IP network, the message transfer is considered safe. To avoid security vulnerabilities from shortcircuit, disable the feature.

## Hadoop data isolation

This topic describes Hadoop data isolation.

Hadoop super users can control the data in the file system. If you do not want the Hadoop super users to access the data in the POSIX applications, configure `gpfs.data.dir` in `/usr/lpp/mmfs/hadoop/etc/hadoop/gpf-site.xml` to isolate the Hadoop data under `/<gpfs.mnt.dir>/<gpfs.data.dir>/`. This configuration setting ensures that the Hadoop super users can only access the data in the `/<gpfs.mnt.dir>/<gpfs.data.dir>/` folder. Data outside the `/<gpfs.mnt.dir>/<gpfs.data.dir>/` folder cannot be accessed by the Hadoop super users.

## Hadoop data access audit

This section describes the Hadoop data access audit.

HDFS Transparency is certified with IBM Security Guardium DAM (Database Activity Monitoring) to monitor the Hadoop data access over IBM Spectrum Scale.

For more information about Hadoop data access audit, see Big data security and auditing with IBM InfoSphere Guardium.

# Guide for security setup

## Enable Kerberos for IBM BigInsights IOP

**For manual HDFS replacement mode:**

This topic lists the steps to enable Kerberos for IBM BigInsights IOP for manual HDFS replacement mode.

In this mode, users must install IOP over HDFS, and then replace HDFS with IBM Spectrum Scale.

If you use this mode to deploy IOP and IBM Spectrum Scale, perform the following steps to enable Kerberos:

1. To set up the KDC server, go to Setting up a KDC manually.
2. Shut down the GPFS service and start the HDFS service.

   **Note:** For the FPO model and the shared storage model, HDFS transparency nodes must be part of the IOP cluster. IOP services do not start when the NameNode service is running over the node outside the IOP cluster.
3. In the Ambari GUI, click **Admin** > **Kerberos**, and follow the guide to enable the Kerberos service.

   **Note:** In the GUI wizard, select existing MIT KDC and type the required input according to the configuration.
4. Run a service check for all the services. For a user who has not authenticated with the KDC, the system reports a failure:

```
[fvtest@c8f2n06 ~]$ yarn org.apache.hadoop.yarn.applications.distributedshell.Client
-shell_command ls -num_containers 3 -jar
/usr/iop/current/hadoop-yarn-client/hadoop-yarn-applications-distributedshell.jar
15/11/02 02:51:46 INFO distributedshell.Client: Initializing Client
15/11/02 02:51:46 INFO distributedshell.Client: Running Client
15/11/02 02:51:46 INFO client.RMProxy: Connecting to ResourceManager at
c8f2n07.gpfs.net/192.168.105.163:8050
15/11/02 02:51:47 WARN ipc.Client: Exception encountered while connecting to the
server : javax.security.sasl.SaslException: GSS initiate failed [Caused by GSSException:
No valid credentials provided (Mechanism level: Failed to find any Kerberos tgt)]
15/11/02 02:51:47 FATAL distributedshell.Client: Error running Client
java.io.IOException: Failed on local exception: java.io.IOException:
javax.security.sasl.SaslException: GSS initiate failed
[Caused by GSSException: No valid credentials provided (Mechanism level:
Failed to find any Kerberos tgt)]; Host Details : local host is:
"c8f2n06/192.168.105.162"; destination host is: "c8f2n07.gpfs.net":8050;
        at org.apache.hadoop.net.NetUtils.wrapException(NetUtils.java:773)
        at org.apache.hadoop.ipc.Client.call(Client.java:1480)
        at org.apache.hadoop.ipc.Client.call(Client.java:1407)
        at org.apache.hadoop.ipc.ProtobufRpcEngine$Invoker.invoke(ProtobufRpcEngine.java:229)
        at com.sun.proxy.$Proxy7.getClusterMetrics(Unknown Source)
        at org.apache.hadoop.yarn.api.impl.pb.client.ApplicationClientProtocolPBClientImpl.getClusterMetrics
        (ApplicationClientProtocolPBClientImpl.java:206)
```

5. Shut down the HDFS service.
6. In Ambari GUI, click **HDFS** > **configs** in the Advanced tag page, and add the following to Custom core-site:

```
hadoop.proxyuser.yarn.groups=*
hadoop.proxyuser.yarn.hosts=*
```

   If HTTPS is not configured, the user must ensure that `dfs.http.policy` is HTTP_ONLY and `dfs.https.enable` is false. Otherwise, the IOP services will not start.
7. Install the gpfs.hdfs-protocol rpm.
8. On any one IOP node, run the **mmhadoopctl connector syncconf /etc/hadoop/conf** command. This synchronizes all the Hadoop configurations into the HDFS transparency configuration directory.

**Note:** The **/etc/hadoop/conf** parameter is the Hadoop configuration directory for IOP.

9. Modify `/usr/lpp/mmfs/hadoop/etc/hadoop/gpfs-site.xml` according to your cluster.
10. On the same node, run:

    ```
    /usr/lpp/mmfs/hadoop/sbin/deploy-gpfs.sh –nocheck /usr/lpp/mmfs/hadoop/etc/hadoop/
    /usr/lpp/mmfs/hadoop/etc/hadoop/
    ```

11. For an FPO model, run the following commands on any one node:

    ```
    mmdsh –N all "chown root:root /etc/security/keytabs/dn.service.keytab"
    mmdsh -N all "chown root:root /var/lib/hadoop-hdfs"
    ```

    For the shared storage model, run the following commands on any one node that is running the HDFS transparency service:

    ```
    mmdsh –N all "chown root:root /etc/security/keytabs/dn.service.keytab"
    mmdsh -N all "chown root:root /var/lib/hadoop-hdfs"
    ```

    The `/usr/lpp/mmfs/hadoop/sbin/mmhadoopctl` connector starts.
12. Go to the IOP Ambari GUI to start the other services.

    **Note:** If shortcircuit is enabled, `dfs.domain.socket.path=/var/lib/hadoop-hdfs/dn_socket` must be owned by `root:root`. If it is not, the DataNode service will not start.

**Automatic GPFS deployment with IOP:**

In Release gpfs.hdfs-protocol.2.7.0-1, HDFS transparency is not integrated with BigInsights IOP.

**Enable the HTTPS service of NameNode:**

This topic lists the steps to enable the HTTPS service.

By default, the HTTPS service is not enabled. If you want to enable the HTTPS service, perform the following steps:

1. Generate the key and the certificate. To deploy HTTPS, a key and certificate must be generated for each machine in the cluster. You can use Java's keytool utility to accomplish this task:

   ```
   $ keytool -keystore {keystore} -alias localhost -validity {validity} –genkey
   ```

   The parameter definitions are as following:

   - keystore: The keystore file that stores the certificate. The keystore file contains the private key of the certificate, and must be kept safely.
   - validity: The validity of the certificate in days.

   The keytool needs more details of the certificate, such as the hostname and the organization name.

   **Note:** The hostname (CN) is the hostname of the HDFS Transparency NameNode.
2. Create your own CA. Each machine in the cluster has a public-private key pair, and a certificate to identify the machine. The certificate, however, is unsigned, which means that an attacker can create such a certificate and become an authorized user. Use `openssl` to generate a new CA certificate:

   ```
   openssl req -new -x509 -keyout <ca-key> -out <ca-cert> -days <validity>
   ```

   The generated CA is simply a public-private key pair and certificate and is intended to sign other certificates.
3. Add the generated CA to the client truststore.

   ```
   $ keytool -keystore {truststore} -alias CARoot -import -file {ca-cert}
   ```

   In contrast to the keystore that stores the machine identity, the truststore of the client stores all the certificates that the client must trust. Importing a certificate into a truststore means that the client trusts all the certificates that are signed by that certificate. This attribute is called the chain of trust, and it is particularly useful while deploying HTTPS on a large Hadoop cluster. You can sign all the certificates in the cluster with a single CA, and have all machines share the same truststore that trusts the CA. The machines can then authenticate all other machines.

4. Sign all the generated certificates and the CA. Perform the following steps:
   a. Export the certificate from the keystore.
      ```
      $ keytool -keystore -alias localhost -certreq -file {cert-file}
      ```
   b. Sign the certificate with the CA.
      ```
      $ openssl x509 -req -CA {ca-cert} -CAkey {ca-key} -in {cert-file}
      -out {cert-signed} -days {validity} -CAcreateserial -passin pass:{ca-password}
      ```
   c. Import the CA certificate and the signed certificate into the keystore.
      ```
      $ keytool -keystore -alias CARoot -import -file {ca-cert}
      $ keytool -keystore -alias localhost -import -file {cert-signed}
      ```

   The parameter definitions are as follows:
   - keystore: the location of the keystore
   - ca-cert: the certificate of the CA
   - ca-key: the private key of the CA
   - ca-password: the passphrase of the CA
   - cert-file: the exported, unsigned certificate of the server
   - cert-signed: the signed certificate of the server
5. Configure the HDFS transparency NameNode in the hdfs-site.xml file:
   ```
   <property>
   <name>dfs.http.policy</name>
   <value>HTTP_AND_HTTPS</value>
   </property>

   <property>
   <name>dfs.https.enable</name>
   <value>true</value>
   </property>
   ```

   The **dfs.http.policy** parameter can be one of the following:
   - HTTP_ONLY: Only the HTTP server has started.
   - HTTPS_ONLY: Only the HTTPS server has started.
   - HTTP_AND_HTTPS: The HTTP and HTTPS servers have started.

   **Note:** If you configure the **dfs.http.policy** parameter as HTTPS_ONLY or HTTP_AND_HTTPS, **webhdfs** of HDFS transparency NameNode becomes unavailable. For more information about applications requiring **swebhdfs**, see HDFS-3987.

   **Note:** The **swebhdfs** parameter is not available for Hadoop Release 2.3.0 and earlier. You must consider upgrading your Hadoop release version for enhanced security.
6. Configure ssl-server.xml as:
   ```
   <property>
   <name>ssl.server.keystore.type</name>
   <value>jks</value>
   </property>
   <property>
   <name>ssl.server.keystore.keypassword</name>
   <value><password of keystore></value>
   </property>
   <property>
   <name>ssl.server.keystore.location</name>
   <value><location of keystore.jks></value>
   </property>
   <property>
   <name>ssl.server.truststore.type</name>
   <value>jks</value>
   </property>
   <property>
   <name>ssl.server.truststore.location</name>
   ```

```
<value><location of truststore.jks></value>
</property>
<property>
<name>ssl.server.truststore.password</name>
<value><password of truststore></value>
</property>
```

Also, configure the ssl-client.xml as:

```
<property>
<name>ssl.client.truststore.password</name>
<value><password of truststore></value>
</property>

<property>
<name>ssl.client.truststore.type</name>
<value>jks</value>
</property>
<property>
<name>ssl.client.truststore.location</name>
<value><location of truststore.jks></value>
</property>
```

7. To restart the HDFS transparency services, run the **mmhadoopctl** command.

   **Note:** Remember to sync hdfs-site.xml, ssl-server.xml, and ssl-client.xml from BigInsights IOP, `/etc/hadoop/conf`, with the HDFS transparency configuration directory, `/usr/lpp/mmfs/hadoop/etc/hadoop`, for all nodes that are running HDFS transparency.

## Security configuration in Hadoop

This topic describes the security configuration in Hadoop.

The Kerberos-related configuration changes in the hdfs-site.xml file are restricted to HDFS transparency. However, enabling Kerberos impacts the other Hadoop components as well. Therefore, other components must also be configured for Kerberos.

If changes are made only to the hdfs-site.xml file, which is the configuration file used by HDFS transparency, the other Hadoop services fail.

Add the following configuration settings to the core-site.xml file:

```
<property>
 <name>hadoop.http.authentication.cookie.domain</name>
 <value></value>
</property>

<property>
 <name>hadoop.http.authentication.cookie.path</name>
 <value></value>
</property>

<property>
 <name>hadoop.http.authentication.kerberos.name.rules</name>
 <value></value>
</property>

<property>
 <name>hadoop.http.authentication.signature.secret</name>
 <value></value>
</property>

<property>
 <name>hadoop.http.authentication.signature.secret.file</name>
 <value></value>
</property>
```

```
  <property>
   <name>hadoop.http.authentication.signer.secret.provider</name>
   <value></value>
  </property>

  <property>
   <name>hadoop.http.authentication.signer.secret.provider.object</name>
   <value></value>
  </property>

  <property>
   <name>hadoop.http.authentication.token.validity</name>
   <value></value>
  </property>

  <property>
   <name>hadoop.http.authentication.type</name>
   <value>simple</value>
  </property>

  <property>
   <name>hadoop.http.filter.initializers</name>
   <value></value>
  </property>

  <property>
   <name>hadoop.proxyuser.HTTP.groups</name>
   <value>users</value>
  </property>

  <property>
   <name>hadoop.proxyuser.HTTP.hosts</name>
   <value>c8f2n07.gpfs.net</value>
  </property>

  <property>
   <name>hadoop.proxyuser.knox.groups</name>
   <value>users</value>
  </property>

  <property>
   <name>hadoop.proxyuser.knox.hosts</name>
   <value>c8f2n06.gpfs.net</value>
  </property>

  <property>
   <name>hadoop.rpc.protection</name>
   <value>authentication</value>
  </property>

  <property>
   <name>hadoop.security.auth_to_local</name>
   <value>RULE:[1:$1@$0](ambari-qa@gpfs.net)s/.*/ambari-qa/
RULE:[1:$1@$0](hbase@gpfs.net)s/.*/hbase/
RULE:[1:$1@$0](hdfs@gpfs.net)s/.*/hdfs/
RULE:[1:$1@$0](spark-08212047@gpfs.net)s/.*/spark/
RULE:[1:$1@$0](.*@gpfs.net)s/@.*//
RULE:[2:$1@$0](HTTP@gpfs.net)s/.*/hbase/
RULE:[2:$1@$0](amshbase@gpfs.net)s/.*/ams/
RULE:[2:$1@$0](dn@gpfs.net)s/.*/hdfs/
RULE:[2:$1@$0](hbase@gpfs.net)s/.*/hbase/
RULE:[2:$1@$0](hive@gpfs.net)s/.*/hive/
RULE:[2:$1@$0](jhs@gpfs.net)s/.*/mapred/
RULE:[2:$1@$0](jn@gpfs.net)s/.*/hdfs/
RULE:[2:$1@$0](knox@gpfs.net)s/.*/knox/
RULE:[2:$1@$0](nfs@gpfs.net)s/.*/hdfs/
RULE:[2:$1@$0](nm@gpfs.net)s/.*/yarn/
```

```
RULE:[2:$1@$0](nn@gpfs.net)s/.*/hdfs/
RULE:[2:$1@$0](oozie@gpfs.net)s/.*/oozie/
RULE:[2:$1@$0](rm@gpfs.net)s/.*/yarn/
RULE:[2:$1@$0](solr@gpfs.net)s/.*/solr/
RULE:[2:$1@$0](yarn@gpfs.net)s/.*/yarn/
RULE:[2:$1@$0](zookeeper@gpfs.net)s/.*/ams/
RULE:[2:$1@$0]([nd]n@.*)s/.*/hdfs/
RULE:[2:$1@$0]([rn]m@.*)s/.*/yarn/
RULE:[2:$1@$0](hm@.*)s/.*/hbase/
RULE:[2:$1@$0](jhs@.*)s/.*/mapred/
RULE:[2:$1@$0](rs@.*)s/.*/hbase/
DEFAULT</value>
  </property>

  <property>
   <name>hadoop.security.authentication</name>
   <value>kerberos</value>
  </property>

  <property>
   <name>hadoop.security.authorization</name>
   <value>true</value>
  </property>
```

Add the following configuration settings in the hdfs-site.xml file. Even if there is a single property, modify the property:

```
  <property>
   <name>dfs.datanode.address</name>
   <value>0.0.0.0:1019</value>
  </property>

  <property>
   <name>dfs.datanode.http.address</name>
   <value>0.0.0.0:1022</value>
  </property>

  <property>
   <name>dfs.datanode.kerberos.principal</name>
   <value>dn/_HOST@gpfs.net</value>
  </property>

  <property>
   <name>dfs.namenode.kerberos.internal.spnego.principal</name>
   <value>HTTP/_HOST@gpfs.net</value>
  </property>

  <property>
   <name>dfs.namenode.kerberos.principal</name>
   <value>nn/_HOST@gpfs.net</value>
  </property>

  <property>
   <name>dfs.secondary.namenode.kerberos.internal.spnego.principal</name>
   <value>HTTP/_HOST@gpfs.net</value>
  </property>

  <property>
   <name>dfs.secondary.namenode.kerberos.principal</name>
   <value>nn/_HOST@gpfs.net</value>
  </property>

  <property>
   <name>dfs.web.authentication.kerberos.principal</name>
   <value>HTTP/_HOST@gpfs.net</value>
  </property>
```

```
<property>
 <name>nfs.kerberos.principal</name>
 <value>nfs/_HOST@gpfs.net</value>
</property>

<property>
 <name>nfs.keytab.file</name>
 <value>/etc/security/keytabs/nfs.service.keytab</value>
</property>

<property>
 <name>dfs.http.policy</name>
 <value>HTTP_AND_HTTPS</value>
</property>

<property>
 <name>dfs.https.enable</name>
 <value>true</value>
</property>
```

The following is the modification applied to the mapred-site.xml file:

**Note:** If the property exists, modify the property.

```
<property>
 <name>mapreduce.jobhistory.keytab</name>
 <value>/etc/security/keytabs/jhs.service.keytab</value>
</property>

<property>
 <name>mapreduce.jobhistory.principal</name>
 <value>jhs/_HOST@gpfs.net</value>
</property>

<property>
 <name>mapreduce.jobhistory.webapp.spnego-keytab-file</name>
 <value>/etc/security/keytabs/spnego.service.keytab</value>
</property>

<property>
 <name>mapreduce.jobhistory.webapp.spnego-principal</name>
 <value>HTTP/_HOST@gpfs.net</value>
</property>
```

The following is the modification applied to the yarn-site.xml file:

**Note:** If the property exists, modify the property.

```
<property>
 <name>yarn.acl.enable</name>
 <value>true</value>
</property>

<property>
 <name>yarn.nodemanager.container-executor.class</name>
 <value>org.apache.hadoop.yarn.server.nodemanager.LinuxContainerExecutor</value>
</property>

<property>
 <name>yarn.nodemanager.keytab</name>
 <value>/etc/security/keytabs/nm.service.keytab</value>
</property>

<property>
 <name>yarn.nodemanager.linux-container-executor.cgroups.mount-path</name>
 <value></value>
</property>
```

```
<property>
 <name>yarn.nodemanager.principal</name>
 <value>nm/_HOST@gpfs.net</value>
</property>

<property>
 <name>yarn.nodemanager.webapp.spnego-keytab-file</name>
 <value>/etc/security/keytabs/spnego.service.keytab</value>
</property>

<property>
 <name>yarn.nodemanager.webapp.spnego-principal</name>
 <value>HTTP/_HOST@gpfs.net</value>
</property>

<property>
 <name>yarn.resourcemanager.keytab</name>
 <value>/etc/security/keytabs/rm.service.keytab</value>
</property>

<property>
 <name>yarn.resourcemanager.principal</name>
 <value>rm/_HOST@gpfs.net</value>
</property>

<property>
 <name>yarn.resourcemanager.proxy-user-privileges.enabled</name>
 <value>true</value>
</property>

<property>
 <name>yarn.resourcemanager.proxyusers.*.groups</name>
 <value></value>
</property>

<property>
 <name>yarn.resourcemanager.proxyusers.*.hosts</name>
 <value></value>
</property>

<property>
 <name>yarn.resourcemanager.proxyusers.*.users</name>
 <value></value>
</property>

<property>
 <name>yarn.resourcemanager.webapp.spnego-keytab-file</name>
 <value>/etc/security/keytabs/spnego.service.keytab</value>
</property>

<property>
 <name>yarn.resourcemanager.webapp.spnego-principal</name>
 <value>HTTP/_HOST@gpfs.net</value>
</property>

<property>
 <name>yarn.timeline-service.enabled</name>
 <value>false</value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.cookie.domain</name>
 <value></value>
</property>

<property>
```

```
 <name>yarn.timeline-service.http-authentication.cookie.path</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.kerberos.keytab</name>
 <value>/etc/security/keytabs/spnego.service.keytab</value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.kerberos.name.rules</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.kerberos.principal</name>
 <value>HTTP/_HOST@gpfs.net</value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.proxyusers.*.groups</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.proxyusers.*.hosts</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.proxyusers.*.users</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.signature.secret</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.signature.secret.file</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.signer.secret.provider</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.signer.secret.provider.object</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.token.validity</name>
 <value></value>
</property>

<property>
 <name>yarn.timeline-service.http-authentication.type</name>
 <value>kerberos</value>
</property>

<property>
 <name>yarn.timeline-service.keytab</name>
 <value>/etc/security/keytabs/yarn.service.keytab</value>
```

```
</property>

<property>
 <name>yarn.timeline-service.principal</name>
 <value>yarn/_HOST@gpfs.net</value>
</property>
```

# Chapter 33. Encryption

GPFS provides support for file encryption that ensures both secure storage and secure deletion of data. GPFS manages encryption through the use of encryption keys and encryption policies.

**Note:** GPFS encryption is only available with IBM Spectrum Scale Advanced Edition. The file system must be at GPFS V4.1 or later. Encryption is supported in multicluster environments (provided that the remote nodes have their own `/var/mmfs/etc/RKM.conf` files and access to the remote key management servers; see "Encryption keys") and FPO environments.

Secure storage uses encryption to make data unreadable to anyone who does not possess the necessary encryption keys. The data is encrypted while "at rest" (on disk) and is decrypted on the way to the reader. Only data, not metadata, is encrypted.

GPFS encryption can protect against attacks targeting the disks (for example, theft or acquisition of improperly discarded disks) as well as attacks performed by unprivileged users of a GPFS node in a multi-tenant cluster (that is, a cluster that stores data belonging to multiple administrative entities called tenants). However, it cannot protect against deliberate malicious acts by a cluster administrator.

Secure data deletion leverages encryption and key management to guarantee erasure of files beyond the physical and logical limitations of normal deletion operations. If data is encrypted, and the master key (or keys) required to decrypt it have been deleted from the key server, that data is effectively no longer retrievable. See "Encryption keys."

**Important:** Encryption should not be viewed as a substitute for using file permissions to control user access.

For more information on configuring encryption, see the Encryption chapter in the IBM Spectrum Scale Redbook.

## Encryption keys

GPFS uses the following types of encryption keys:

**master encryption key (MEK)**

An MEK is used to encrypt file encryption keys.

MEKs are stored in remote key management (RKM) servers and are cached by GPFS components. GPFS receives information about the RKM servers in a separate `/var/mmfs/etc/RKM.conf` configuration file. Encryption rules present in the encryption policy define which MEKs should be used, and the `/var/mmfs/etc/RKM.conf` file provides a means of accessing those keys. The `/var/mmfs/etc/RKM.conf` also specifies how to access RKMs containing MEKs used to encrypt files created under previous encryption policies.

An MEK is identified with a unique *Keyname* that combines the name of the key and the RKM server on which it resides. See "Encryption policy rules" on page 538 for *Keyname* format.

**file encryption key (FEK)**

An FEK is used to encrypt sectors of an individual file. It is a unique key that is randomly generated when the file is created. For protection, it is encrypted (or "wrapped") with one or more MEKs and stored in the `gpfs.Encryption` extended attribute of the file.

A wrapped FEK cannot be decoded without access to the MEK (or MEKs) used to wrap it. Therefore, a wrapped FEK is useless to an attacker and does not require any special handling at

object deletion time. If necessary, an FEK can be rewrapped using a new set of MEKs to allow for operations like MEK expiration and rotation, compromised key removal, and data expiration.

**Note:** If an encryption policy specifies that an FEK be wrapped multiple times, only one of the wrapped-FEK instances needs to be unwrapped for the file to be accessible.

## Encryption policies

GPFS uses encryption policies to manage aspects of how file encryption is to be implemented, including the following:

- which files are to be encrypted
- which algorithm is to be used for the encryption
- which MEK (or MEKs) are to be used to wrap the FEK of a file

Encryption policies are configured using the **mmchpolicy** command and are applied at file creation time. When a file is created, encryption rules are traversed in order until one of the following occurs:

- The last rule is reached.
- The maximum number of **SET ENCRYPTION** rules that can be matched (eight) is reached.
- An **ENCRYPTION EXCLUDE** rule is matched.

If the file matches at least one **SET ENCRYPTION** rule, an FEK is generated and used to encrypt its contents. The FEK is wrapped once for each policy it matches, resulting in one or more versions of the encrypted FEK being stored in the `gpfs.Encryption` extended attribute of the file.

**Notes:**

1. When an encryption policy is changed, the changes apply only to the encryption of subsequently created files.
2. Encryption policies are defined on a per–file system basis by a system administrator. Once the encryption policies are put in place, they may result in files in different filesets or with different names being encrypted differently.

## Encryption policy rules

GPFS provides the following rules with which you can specify encryption policies:

**ENCRYPTION IS**

This rule is used to specify how a file is to be encrypted and how the FEK is to be wrapped.

The syntax of the **ENCRYPTION IS** rule is:

```
RULE 'RuleName' ENCRYPTION 'EncryptionSpecificationName' IS
    ALGO 'EncParamString'
    COMBINE 'CombineParamString'
    WRAP 'WrapParamString'
    KEYS('Keyname'[, 'Keyname', ... ])
```

where:

**ALGO** *EncParamString*

specifies the encryption parameter string, which defines the following:

- encryption algorithm
- key length
- mode of operation
- key derivation function

The following encryption parameter strings are valid:

*Table 49. Valid EncParamString values*

| Value | Description |
|---|---|
| AES:128:XTS:FEK:HMACSHA512 | Encrypt the file with AES in XTS mode. The FEK is 128 bits long and is preprocessed using HMAC with SHA-512. |
| AES:256:XTS:FEK:HMACSHA512 | Encrypt the file with AES in XTS mode. The FEK is 256 bits long and is preprocessed using HMAC with SHA-512. |
| AES:128:CBC:FEK:HMACSHA512 | Encrypt the file with AES in CBC mode. The FEK is 128 bits long and is preprocessed using HMAC with SHA-512. |
| AES:192:CBC:FEK:HMACSHA512 | Encrypt the file with AES in CBC mode. The FEK is 192 bits long and is preprocessed using HMAC with SHA-512. |
| AES:256:CBC:FEK:HMACSHA512 | Encrypt the file with AES in CBC mode. The FEK is 256 bits long and is preprocessed using HMAC with SHA-512. |

**COMBINE** *CombineParamString*
specifies a string that defines the mode to be used to combine MEKs specified by the **KEY** statement.

The following combine parameter string values are valid:

*Table 50. Valid combine parameter string values*

| Value | Description |
|---|---|
| XORHMACSHA512 | Combine MEKs with a round of XOR followed by a round of HMAC with SHA-512. |
| XOR | Combine MEKs with a round of XOR. |

**WRAP** *WrapParamString*
specifies a string that defines the encryption algorithm and the wrapping mode to be used to wrap the FEK.

The following wrapping parameter string values are valid:

*Table 51. Valid wrapping parameter string values.*

| Value | Description |
|---|---|
| AES:KWRAP | Use AES key wrap to wrap the FEK. |
| AES:CBCIV | Use AES in CBC-IV mode to wrap the FEK. |

**KEYS ('***Keyname***'[, '***Keyname***', ... ])**
specifies one or more keys to be applied. Each *Keyname* is a unique identifier that combines the name of the key and the RKM server on which it resides. The format for *Keyname* is:

*KeyId*:*RkmId*

where

*KeyId*
An internal identifier that uniquely identifies the key inside the RKM. Valid characters for *KeyId* are the following: 'A' through 'Z'; 'a' through 'z'; '0' through '9'; and '-' (hyphen). The minimum length of *KeyId* is one character; the maximum length is 42 characters.

*RkmId*
The identifier of the `/var/mmfs/etc/RKM.conf` entry for the RKM that manages the key. An RKM ID must be unique within the cluster, must be 1-21 characters in length, and can contain only the characters a - z, A - Z, 0 - 9, or underscore (_). The first character cannot be a numeral.

**Notes:**

1. The maximum number of keys you can specify with the **ENCRYPTION IS** rule is eight.
2. The number of keys that can be used to encrypt a single file is permanently limited by the inode size of the file system.
3. You cannot specify the same key more than once in a given **ENCRYPTION IS** rule. Also, do not specify keys with identical values in an **ENCRYPTION IS** rule. Specifying the same key or identically-valued keys could result in a security breach for your data.

**SET ENCRYPTION**
The **SET ENCRYPTION** rule is similar to the **SET POOL** rule. If more than one such rule is present, all **SET ENCRYPTION** rules are considered and the FEK is wrapped once for each of the rules that apply (up to the maximum of eight). As mentioned in "Encryption keys" on page 537, if an FEK is wrapped multiple times, only one of the wrapped-FEK instances needs to be unwrapped for the file to be accessed.

If no **SET ENCRYPTION** rule is applicable at create time, the file is not encrypted.
The syntax of the **SET ENCRYPTION** rule is:

```
RULE 'RuleName' SET ENCRYPTION 'EncryptionSpecificationName'[, 'EncryptionSpecificationName',...]
    [FOR FILESET ('FilesetName'[,'FilesetName']...)]
    [WHERE SqlExpression]
```

where:

*EncryptionSpecificationName*
is the name of a specification defined by an **ENCRYPTION IS** rule.

To stop traversing policy rules at a certain point and encrypt using only those rules that have matched up to that point, use the **SET ENCRYPTION EXCLUDE** rule:

```
RULE ['RuleName']  SET ENCRYPTION EXCLUDE
     [FOR FILESET ('FilesetName'[,'FilesetName']...)]
     [WHERE SqlExpression]
```

**Note:** Encryption policies do not support the **ACTION** clause.

## Default encryption parameters

To simplify policy management, GPFS accepts the special default value `'DEFAULTNISTSP800131A'` as the **ALGO** parameter string.

For example, this policy statement:

```
RULE 'somerule' ENCRYPTION 'somename' IS
    ALGO 'DEFAULTNISTSP800131A'
    KEYS('KEY-2f1f7700-de74-4e55-a9be-bee49c5b3af8:RKMKMIP3')
```

corresponds to this:

```
RULE 'somerule' ENCRYPTION 'somename' IS
    ALGO 'AES:256:XTS:FEK:HMACSHA512'
    COMBINE 'XORHMACSHA512'
    WRAP 'AES:KWRAP'
    KEYS('KEY-2f1f7700-de74-4e55-a9be-bee49c5b3af8:RKMKMIP3')
```

**Note:** When this special **ALGO** default value is set as the **ALGO** *EncParamString*, neither **COMBINE** nor **WRAP** should be specified.

## Example of an encryption policy

This is an example of an encryption policy:

```
RULE 'myEncRule1' ENCRYPTION 'E1' IS
         ALGO 'DEFAULTNISTSP800131A'
         KEYS('1:RKM_1', '2:RKM_2')

RULE 'myEncRule2' ENCRYPTION 'E2' IS
         ALGO 'AES:256:XTS:FEK:HMACSHA512'
         COMBINE 'XOR'
         WRAP 'AES:KWRAP'
         KEYS('3:RKM_1')

RULE 'myEncRule3' ENCRYPTION 'E3' IS
         ALGO 'AES:128:CBC:FEK:HMACSHA512'
         COMBINE 'XORHMACSHA512'
         WRAP 'AES:CBCIV'
         KEYS('4:RKM_2')

RULE 'Do not encrypt files with extension enc4'
         SET ENCRYPTION EXCLUDE
         FOR FILESET('fs1')
         WHERE NAME LIKE '%.enc4'

RULE 'Encrypt files with extension enc1 with rule E1'
         SET ENCRYPTION 'E1'
         FOR FILESET('fs1')
         WHERE NAME LIKE '%.enc1'

RULE 'Encrypt files with extension enc2 with rule E2'
         SET ENCRYPTION 'E2'
         FOR FILESET('fs1')
         WHERE NAME LIKE '%.enc2'

RULE 'Encrypt files with extension enc* with rule E3'
         SET ENCRYPTION 'E3'
         FOR FILESET('fs1')
         WHERE NAME LIKE '%.enc%'
```

**Note:**

In this example encryption policy:

- All files in fileset `fs1` are treated as follows:
  - If the extension is equal to `enc4`, the file is not encrypted. This happens because the `ENCRYPTION EXCLUDE` rule is matched first, stopping the traversal of the remaining rules before any additional matches can be made.
  - If the extension is equal to `enc1`, the file is encrypted with a 256-bit FEK, using AES in XTS mode; the FEK is preprocessed with HMAC with SHA-512, and the FEK is then wrapped twice:
    - once with AES key wrap, with keys `1:RKM_1` and `2:RKM_2` combined via one round of XOR followed by one round of HMAC with SHA-512
    - once with AES in CBC-IV mode using key `4:RKM_2`

    This happens because both rules `E1` and `E3` apply, since extension `enc1` matches both `%.enc1` and `%.enc%`. Note that the encryption algorithms specified by rule `E1`, which grant a stronger security than those of rule `E3`, are chosen and applied.
  - If the extension is equal to `enc2`, the file is encrypted with a 256-bit FEK, using AES in XTS mode; the FEK is preprocessed with HMAC with SHA-512; and the FEK is then wrapped twice:

- once with AES key wrap using key 3:RKM_1
- once with AES in CBC-IV mode using key 4:RKM_2

This happens because both rules E2 and E3 apply, since extension enc2 matches both %.enc2 and %.enc%.

– If the extension is equal to enc3, the file is encrypted with a 128-bit FEK, using AES in CBC mode; the FEK is preprocessed with HMAC with SHA-512; and the FEK is then wrapped once with AES in CBC-IV mode using key 4:RKM_2.

This happens because only rule E3 applies, since extension enc3 only matches %.enc%.

- A GPFS node with access to both keys 1:RKM_1 and 2:RKM_2 or to key 4:RKM_2 can access a file with extension enc1.
- A GPFS node with access to key 3:RKM_1 or to key 4:RKM_2 can access a file with extension enc2.
- A GPFS node with access to key 4:RKM_2 can access a file with extension enc3.
- No key is required to access a file with extension enc4.
- A file with extension enc1 is securely deleted when either key 1:RKM_1 or 2:RKM_2, and key 4:RKM_2 are destroyed in their respective RKMs (and their cached copies have been flushed).
- A file with extension enc2 is securely deleted when key 3:RKM_1 and key 4:RKM_2 are destroyed in their respective RKMs (and their cached copies have been flushed).
- A file with extension enc3 is securely deleted when key 4:RKM_2 is destroyed in its respective RKM (and its cached copies have been flushed).
- Once created, a file may not be encrypted with more MEKs, only with different MEKs using the **REWRAP** rule.

## Rewrapping policies

Rewrapping policies are used to change the way a set of FEKs is encrypted; that is, to change the set of MEKs that wrap the FEKs of those files. Rewrapping applies only to files that are already encrypted, and the rewrapping operation acts only on the gpfs.Encryption EA of the files. Rewrapping is done by using the **mmapplypolicy** command to apply a set of policy rules containing one or more CHANGE ENCRYPTION KEYS rules. These rules have the form:

```
RULE 'ruleName' CHANGE ENCRYPTION KEYS FROM 'Keyname_1' to 'Keyname_2'
[FROM POOL 'poolName']
  [FOR FILESET(...)]
  [SHOW(...)]
  [WHERE ... ]
```

where:

- *Keyname_1* is the unique identifier of the MEK to be replaced. (See "Encryption policy rules" on page 538 for *Keyname* format.)
- *Keyname_2* is the unique identifier of the new MEK, which will replace the old MEK identified by *Keyname_1*.
- The FOR FILESET and WHERE clauses narrow down the set of affected files.

Both *Keyname_1* and *Keyname_2* are listed, and only the files that currently use *Keyname_1* will have their FEKs rewrapped with *Keyname_2*. Files that do not currently use *Keyname_1* are not affected by the operation.

**Notes:**

1. Only the *first* matching **CHANGE ENCRYPTION KEYS** rule will be applied to each file. The rule will rewrap each wrapped version of the FEK that was encrypted with the MEK in the **CHANGE ENCRYPTION KEYS** rule.
2. The same MEK cannot be used more than once in a particular wrapping of the FEK.

# Preparation for encryption

Preparing for encryption includes verifying the version of IBM Spectrum Scale, installing a remote encryption key server, preparing the cluster, and preparing the encryption key server back ends.



## Terms defined

You should be familiar with the following terms:

**device group**
: See *tenant*.

**file encryption key**
: A *file encryption key (FEK)* is an encryption key that a key client uses to encrypt a data file. See "Encryption keys" on page 537.

**key client**
: A *key client* is a computer system, such as an IBM Spectrum Scale node, that retrieves master encryption keys from a key server.

**key server**
: A *key server*, also known as a *Remote Key Management (RKM) server*, is a server that provides master encryption keys for key clients. Examples of key server software products are IBM Security Key Lifecycle Manager (SKLM) and Vormetric Data Security Manager (DSM).

**master encryption key**
: A *master encryption key (MEK)* is an encryption key that a key client uses to encrypt a file encryption key. See "Encryption keys" on page 537.

**tenant**
: A *tenant* is an entity on a key server that contains master encryption keys and certificates. In the Vormetric Data Security Manager (DSM), a tenant is called a *device group*.

## Required software: IBM Spectrum Scale

The following table lists the versions of IBM Spectrum Scale that support encryption and the encryption setup methods:

*Table 52. Required version of IBM Spectrum Scale*

| IBM software | Version | Encryption set-up method |
|---|---|---|
| IBM Spectrum Scale | V4.1 or later | Regular setup |
| | V4.2.1 or later | Simplified setup |

## Required software: Remote Key Management (RKM) server

The next table shows the RKM server software that IBM Spectrum Scale supports.

*Table 53. Remote Key Management servers*

| RKM server | Version | Type of encryption setup |
|---|---|---|
| IBM Security Key Lifecycle Manager (SKLM) | V2.5.0.1 or later | Regular setup |
| | V2.5.0.4 or later | Simplified setup |
| Vormetric Data Security Manager (DSM) | V5.2.3 or later | Regular setup |

**Note:** IBM SKLM and Vormetric DSM have a complete implementation of the Key Management Interoperability Protocol (KMIP) standard of the Organization for the Advancement of Structured Information Standards (OASIS). IBM Spectrum Scale nodes use the KMIP protocol to retrieve keys from SKLM and Vormetric Data Security Manager (DSM) servers.

## Preparing your cluster for encryption

Follow these steps:

1. Verify the following items in your IBM Spectrum Scale cluster:
   - The cluster is running the correct version of IBM Spectrum Scale and the correct version of a supported RKM server. These versions are listed in Table 52 on page 543 and Table 53.
   - The file system daemon is running.
2. Ensure that the following packages are installed:
   - `gpfs.gskit`
   - `gpfs.crypto`
3. Set up an IBM Spectrum Scale file system on the cluster. The version of the file system must be IBM Spectrum Scale Release 4.1 or later. Configure the following features on the file system:
   a. Create the file system with the inode size of 4 KB.This size is the recommended minimum size. The 4 KB inode size is recommended to accommodate the gpfs.Encryption extended attribute that is assigned to each encrypted file at file creation time. This extended attribute contains one or more wrapped file encryption keys (FEKs) so it can potentially grow large. For more information, see the help topic.
   b. Enable fast extended attributes. This setting is the default for a newly created file system if you are running V4.1 or later. However, if your file system was migrated from an earlier level, you might need to enter the following command to add support for fast extended attributes:

      **mmmigratefs** *FsName* **--fastea**

## Preparing the remote key management (RKM) server

The preparation of the RKM server depends on the RKM server product that you select and the encryption method that you plan to follow. For more information, see the help topic in the following list that describes the setup of your RKM server:

- "Configuring encryption with SKLM: Simplified setup" on page 547
- "Configuring encryption with SKLM: Regular setup" on page 566
- "Configuring encryption with the Vormetric DSM key server" on page 573

## RKM back ends

The RKM back end includes a local key client, a remote tenant, and an RKM server. A *tenant*, or *device group*, resides on the RKM server and contains master encryption keys that the key client can request. Each RKM back end is described in an RKM stanza in the `RKM.conf` file on the node.

By controlling the contents of this file, the cluster administrator can control which client nodes have access to master encryption keys (MEKs). For example, the same RKM server can be given two different names in `/var/mmfs/etc/RKM.conf` stanzas. Then, the administrator can partition a set of MEKs hosted on a single RKM server into separate subsets of MEKs. These subsets of MEKs might belong to subsets of the nodes of the cluster.

Because the master encryption keys (MEK) are cached in memory, some short-term outages while accessing a key server might not cause issues. However, failure to retrieve the keys might result in errors while creating, opening, reading, or writing files. Although the keys are cached, they are periodically retrieved from the key server to ensure their validity.

To ensure that MEKs are always available, it is recommended that multiple key servers be set up in a high-availability configuration. See the subtopic "Identifying multiple RKM back ends in a high-availability configuration" on page 546.

**Note:** If you are using the simplified method, then the **mmkeyserv** command manages its own `RKM.conf` file and updates it automatically. This includes adding any backup servers for High Availability and other key retrieval properties. The `RKM.conf` file that the **mmkeyserv** command manages is in the `/var/mmfs/ssl/keyServ` directory.

## The `RKM.conf` file and the RKM stanza

The full path of the `RKM.conf` file is `/var/mmfs/etc/RKM.conf`. The following limits apply:

- The length cannot exceed 1 MiB.
- No limit is set on the number of RKM stanzas, if the length limit is not exceeded.

After the file system is configured with encryption policy rules, the file system is considered encrypted. From that point on, each node that has access to that file system must have an `RKM.conf` file present. Otherwise, the file system might not be mounted or might become unmounted.

Each RKM stanza in the `RKM.conf` file describes a connection between a local key client, a remote tenant, and an RKM server. The following code block shows the structure of an RKM stanza:

```
RKM ID {
   type = ISKLM
   kmipServerUri = tls://host:port
   keyStore = /PathToKeyStoreFile
   passphrase = Password
   clientCertLabel = LabelName
   tenantName = NameOfTenant
   [connectionTimeout = ConnectionTimeout]
   [connectionAttempts = ConnectionAttempts]
   [retrySleep = RetrySleepUsec]
}
```

where the terms of the stanza have the following meanings:

**RKM ID**
: The name of the stanza.

**type** `ISKLM` for the regular setup and the simplified setup. `KMIP` for the Vormetric DSM setup.

**kmipServerUri**
: The DNS name or IP address of the SKLM or DSM server and the KMIP SSL port.

**keyStore**
The path and name of the client keystore.

**passphrase**
The password of the client keystore and client certificate.

**clientCertLabel**
The label of the client certificate in the client keystore.

**tenantName**
The name of the tenant or device group.

**connectionTimeout**
The connection timeout, in seconds. The default is 60 seconds. The valid range is 1 - 120 seconds.

**connectionAttempts**
The number of connection attempts. The default is 3 attempts. The valid range is 1 - 10.

**retrySleep**
The retry sleep time, in microseconds. The default is 100,000 (0.1 seconds). The valid range is 1 - 10,000,000 microseconds.

## Identifying multiple RKM back ends in a high-availability configuration

The SKLM supports automated replication across multiple nodes for high-availability deployments. To identify multiple RKM back ends in a high-availability configuration, specify any of the following optional parameters:

```
rkmname3 {
...
   kmipServerUri2 = tls://host:port     # TLS connection to clone number 1 to host on port
   kmipServerUri3 = tls://host:port     # TLS connection to clone number 2 to host on port
   kmipServerUri4 = tls://host:port     # TLS connection to clone number 3 to host on port
   kmipServerUri5 = tls://host:port     # TLS connection to clone number 4 to host on port
   kmipServerUri6 = tls://host:port     # TLS connection to clone number 5 to host on port
...
}
```

If at least one backup is configured, whenever key retrieval from the master fails, IBM Spectrum Scale looks in each backup until it finds the MEK. The addition of the URIs for the clone servers is the only required change within IBM Spectrum Scale. All other configuration parameters (certificates, keys, node, and tenant information) do not need to change, because they are also part of the set of information that is replicated. The administrator is responsible for creating and maintaining any backups.

Additionally, setting up SKLM key server clones can help gain some performance advantage by distributing MEK retrieval requests across the different clones in a round-robin fashion. To achieve this result, the administrator must specify different orderings of the server endpoints on different IBM Spectrum Scale nodes in the `/var/mmfs/etc/RKM.conf` file.

For example, if two cloned SKLM servers are available (such as `tls://keysrv.ibm.com:5696` and `tls://keysrv_backup.ibm.com:5696`), half of the nodes in the cluster can have the following content in `/var/mmfs/etc/RKM.conf`:

```
...
   kmipServerUri  = tls://keysrv.ibm.com:5696
   kmipServerUri2 = tls://keysrv_backup.ibm.com:5696
...
```

The other half can use the following content:

```
...
   kmipServerUri  = tls://keysrv_backup.ibm.com:5696
   kmipServerUri2 = tls://keysrv.ibm.com:5696
...
```

## Establishing an encryption-enabled environment

Establishing an encryption-ready environment requires a sequence of activities which depend on the type and version of the RKM, as well as on the version of IBM Spectrum Scale.

Each summary covers a basic setup with a single encrypted fileset. There are currently three supported deployment scenarios:

- ISKLM version 2.5.0.4 or later (including 2.6) and IBM Spectrum Scale 4.2.1 or later
- Vormetric DSM version 5.2.3 or later and IBM Spectrum Scale 4.2.1 or later
- ISKLM version 2.5.0.1 or later and GPFS Advanced Edition V4.1 or later or IBM Spectrum Scale V4.2 or later

## Configuring encryption with SKLM: Simplified setup

The simplified setup with IBM Security Key Lifecycle Manager (SKLM) requires IBM Spectrum Scale Advanced Edition V4.2.1 or later and SKLM V2.5.0.4 or later (including V2.6).

You should be aware of the following items:

The IBM Spectrum Scale node that you are configuring for encryption must have direct network access to the system where the key server is installed.

**SECURITY NOTE:** The contents of the following files are security-sensitive:

- The `RKM.conf` file:
  - For the simplified setup: `/var/mmfs/ssl/keyServ/RKM.conf`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKM.conf`
- The directory for the client keystore:
  - For the simplified setup: `/var/mmfs/ssl/keyServ`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKMcerts`

IBM Spectrum Scale reads the contents of security-sensitive files only if the following conditions are met:

- They are regular files that are owned by the root user.
- They are in the root group.
- They are readable and writable only by the user.

See the permission bits in the following examples:

- For the simplified setup:

```
-rw-------. 1 root root 2454 Mar 20 10:32 /var/mmfs/ssl/keyServ/RKM.conf
drw-------. 2 root root 4096 Mar 20 11:15 /var/mmfs/ssl/keyServ/
-rw-------. 1 root root 3988 Mar 20 11:15 /var/mmfs/ssl/keyServ/keystore_name.p12
```

  **Note:** In the simplified setup, the **mmkeyserv** command sets the permission bits automatically.

- For the regular setup and the Vormetric DSM setup:

```
-rw-------. 1 root root 2446 Mar 20 12:15 /var/mmfs/etc/RKM.conf
drw-------. 2 root root 4096 Mar 20 13:47 /var/mmfs/etc/RKMcerts
-rw-------. 1 root root 3988 Mar 20 13:47 /var/mmfs/etc/RKMcerts/keystore_name.p12
```

**CAUTION:**
**It is a good practice to take the following precautions:**

- **Ensure that the passphrase for the client certificate file is not leaked through other means, such as the shell history.**
- **Take appropriate precautions to ensure that the security-sensitive files are not lost or corrupted. IBM Spectrum Scale does not manage or replicate the files.**

**Important:** The client keystore must be record-locked when the GPFS daemon starts. If the keystore files are stored on an NFS mount, the encryption initialization process can hang. The cause is a bug that affects the way NFS handles record locking. If you encounter this problem, upgrade your version of NFS or store your keystore file on a local file system. If an upgrade is not possible and no local file system is available, use a RAM drive to store the keystore files.

The setup is greatly simplified by the use of the **mmkeyserv** command, which can communicate with and configure the SKLM server from the IBM Spectrum Scale node. The **mmkeyserv** command automates the following tasks:

- Creating and configuring the client credentials of the IBM Spectrum Scale node.
- Creating a device group and master encryption keys for the node on SKLM.
- Creating an RKM stanza in the `RKM.conf` configuration file.
- Retrieving a server certificate from SKLM and storing it in the PKCS#12 keystore of the client.
- Propagating the encryption configuration and credentials to all the nodes in the IBM Spectrum Scale cluster.

See the following subtopics for instructions:

- "Part 1: Installing Security Key Lifecycle Manager"
- "Part 2: Configuring a node for encryption" on page 549

## Part 1: Installing Security Key Lifecycle Manager

Follow the instructions in this subtopic to install and configure the IBM Security Key Lifecycle Manager (SKLM).

1. Install IBM Security Key Lifecycle Manager version 2.5.0.1 or later. For the installation, choose a system that the IBM Spectrum Scale node that you want to configure has direct network access to. For information about installing SKLM, see the IBM *Security Key Lifecycle Manager: Installation and Configuration Guide (SC27-5335)*. SKLM requires IBM WebSphere Application Server.
2. Configure SKLM to have the same FIPS 140-2 (FIPS) setting as the IBM Spectrum Scale cluster. Follow these steps:
   a. Determine the FIPS setting of the cluster by entering the following command on the command line:

      ```
      mlsconfig FIPS1402mode
      ```

      The command returns `yes` if the cluster complies with FIPS or `no` if not.
   b. On the SKLM server system, open the `SLKMConfig.properties` file.

      **Note:** The location of the `SLKMConfig.properties` file depends on the operating system:
      - On AIX, Linux, and similar operating systems:

        ```
        /opt/IBM/WebSphere/AppServer/products/sklm/config/SKLMConfig.properties
        ```
      - On Microsoft Windows:

        ```
        /opt/IBM/WebSphere/AppServer/products/sklm/config/SKLMConfig.properties
        ```
   c. Add or remove the following line from the `SKLMConfig.properties` file. Add the line to configure SKLM to comply with FIPS, or remove it to have SKLM not comply with FIPS.

      ```
      fips=on
      ```

3. Configure the SKLM server and the WebSphere Application Server to have the same NIST SP800-131a (NIST) setting as the IBM Spectrum Scale cluster. Follow these steps:

   **Note:** To configure NIST compliance in WebSphere Application Server, see the topic, "Transitioning WebSphere Application Server to the SP800-131 security standard" in the product documentation.

   a. Determine the NIST setting of the cluster by entering the following command on the command line:

      ```
      mlsconfig nistCompliance
      ```

      The command returns `SP800-131A` if the cluster complies with NIST or `off` if not.

   b. On the SKLM server system, open the `SLKMConfig.properties` file. For the location of this file, see the note in Step 2.

   c. Add the following line to configure SKLM to comply with NIST or remove it to configure SKLM not to comply with NIST:

      ```
      TransportListener.ssl.protocols=TLSv1.2
      ```

   d. For all V2.5.0.x versions of SKLM, if you are configuring SKLM to comply with NIST, modify the following variable to include only cipher suites that are approved by NIST. The following statement is all on one line:

      ```
      TransportListener.ssl.ciphersuites=TLS_RSA_WITH_AES_256_CBC_SHA256,
          TLS_RSA_WITH_AES_128_CBC_SHA256
      ```

4. SKLM 2.6.0.0 has a known issue that causes server certificates always to be signed with SHA1withRSA.To work around the problem, follow these steps:

   a. While the SKLM server is running, in the `SLKMConfig.properties` file, modify the `requireSHA2Signatures` property as follows:

      ```
      requireSHA2Signatures=true
      ```

   b. Do not restart the server.

   c. Generate a new server certificate and set it to be the one in use.

   d. If you restart the server, you must repeat this workaround before you can add a server certificate that is signed other than with SHA1withRSA.

## Part 2: Configuring a node for encryption

Gather the following information:

- The logon password of the `SKLMAdmin` administrator of the SKLM server
- The certificate chain of the SKLM server (optional)

The following table provides an overview of the configuration process:

*Table 54. Configuring a node for encryption*

| Item | Step |
|---|---|
| Verify the direct network connection between the IBM Spectrum Scale node and the SKLM server. | Step 1 |
| Add the SKLM key server to the configuration. | Step 2 |
| Add a tenant to the key server. | Step 3 |
| Create a key client. | Step 4 |
| Register the key client to the tenant. | Step 5 |
| Create a master encryption key in the tenant. | Step 6 |
| Set up an encryption policy in the cluster. | Step 7 |
| Test the encryption policy. | Step 8 |

Follow these steps:

1. Verify that the IBM Spectrum Scale node that you are configuring for encryption has a direct network connection to the system on which the SKLM key server runs.
2. Run the **mmkeyserv server add** command to add the SKLM key server from Part I to the configuration:
   a. Enter the following command:

      ```
      mmkeyserv server add ServerName
      ```

      where *ServerName* is the host name or IP address of the SKLM key server that you want to add. See the example listing in Figure 19.
   b. Enter the password for the SKLM server when prompted.
   c. To view the certificate chain of the SKLM server, enter `view` when prompted.
   d. Verify that the certificates that are displayed have the same contents as the certificates in the chain that you downloaded from SKLM.
   e. Enter `yes` to trust the certificates or `no` to reject them. If you trust the certificates, the command adds the key server object to the configuration. In the following listing, key server `keyserver01` is added:

```
# mmkeyserv server add keyserver01
Enter password for the key 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:              7f0312e7be18efd72c9d8f37dbb832724859ba4bb5827c230e2161473e0753b367ed49d
993505bd23858541475de8e021e0930725abbd3d25b71edc8fc3de20b7c2db5cd4e865f41c7c410c1d710acf222e1c4
5189108e40568ddcbeb21094264da60a1d96711015a7951eb2655363309d790ab44ee7b26adf8385e2c210b8268c5ae
de5f82f268554a6fc22ece6efeee2a6264706e71416a0dbe8c39ceacd86054d7cc34dda4fffea4605c037d321290556
10821af85dd9819a4d7e4baa70c51addcda720d33bc9f8bbde6d292c028b2f525a0275ebea968c26f8f0c4b604719ae
3b04e71ed7a8188cd6adf68764374b29c91df3d101a941bf8b7189485ad72
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:              227b5befe89f2e55ef628da6b50db1ab842095a54e1505655e3d95fee753a7f7554868a
a79b294c503dc34562cf69c2a2012879675883896856 5c0812c4aedbb0543d396646a269c02bf4c5ce5acba4409a10e
ffbd47ca38ce492698e2dcdc8390b9ae3f4a47c23ee3045ff0145218668f35a63edac68201789ed0db6e5c170f5c6db
49769f0b4c9a5f208746e4342294c447793ed087fa0ac762588faf420febeb3fca411e4e725bd46476e1f9f44759a69
6573af5dbbc9553218c7083c80440f2e542bf56cc5cc18156cce05efd6c2e5fea2b886c5c1e262c10af18b13ccf38c3
533ba025b97bbe62f271545b2ab5c1f50c1dca45ce504dfcfc257362e9b43
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
```

*Figure 19. Example listing for `mmkeyserv server add`*

   f. Run the **mmkeyserv server show** command to verify that the key server is added. The following listing shows that `keyserver01` is created:

```
# 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
```

3. Run the **mmkeyserv tenant add** command to add a tenant to the key server. The command creates the tenant on the SKLM server if it does not exist. A *tenant* is an entity on the SKLM server that can contain encryption keys and certificates. SKLM uses the term *device group* instead of *tenant*.
   a. Enter the following command to add tenant `devG1` to key server `keyserver01`. Enter the password for the SKLM server when prompted:

```
# mmkeyserv tenant add devG1 --server keyserver01
Enter password for the key server keyserver01:
```

   b. Run the **mmkeyserv tenant show** command to verify that the tenant is added. The following listing shows that tenant `devG1` is added to `keyserver01`:

```
# mmkeyserv tenant show
devG1
    Key Server:      keyserver01.gpfs.net
    Registered Client:  (none)
```

4. Run the **mmkeyserv client create** command to create a key client. A key client can request master encryption keys from a tenant after it is registered to the tenant. The command creates a client keystore on the IBM Spectrum Scale node and puts into it a set of client credentials and the certificate chain of the SKLM server. The directory of the keystore is:

   `/var/mmfs/ssl/keyServ`

   a. Enter the following command to create key client `c1Client1` for key server `keyserver01`. Enter the password for the SKLM server and a passphrase for the new keystore when prompted:

```
# mmkeyserv client create c1Client1 --server keyserver01
Enter password for the key server keyserver01:
Create a pass phrase for keystore:
Confirm your pass phrase:
```

   b. Run the **mmkeyserv client show** command to verify that the key client is created. The following listing shows that key client `c1Client1` is created for remote server keyserver01.gpfs.net:

```
# mmkeyserv client show
c1Client1
        Label:               c1Client1
        Key Server:          keyserver01.gpfs.net
        Tenants:             (none)
```

5. Run the **mmkeyserv client register** command to register the key client with the tenant:

   You must provide a remote key management (RKM) ID as an input for this command. An RKM ID identifies an RKM stanza, which is a structure on the node in which configuration information about the key client, the tenant, and the key server is stored. Each stanza holds the information about one connection between a key client, a tenant, and a key server.

   It is a good practice to use a format like the following one to ensure that the RKM ID is unique:

   *keyServerName_tenantName*

   For example, the RKM ID for the key server and the tenant in these instructions is `keyserver01_devG1`.

   a. Enter the following command to register key client `c1Client1`with tenant `devG1` under RKM ID `keyserver01_devG1`. Enter the requested information when prompted:

```
# mmkeyserv client register c1Client1 --tenant devG1 --rkm-id keyserver01_devG1
Enter password for the key server:

mmkeyserv: [I] Client currently does not have access to the key.  Continue the registration
    process ...
mmkeyserv: Successfully accepted client certificate
```

b. Run the command **mmkeyserv tenant show** to verify that the key client is known to the tenant. The following listing shows that tenant `devG1` lists `c1Client1` as a registered client:

```
mmkeyserv tenant show
devG1
    Key Server:      keyserver01.gpfs.net
    Registered Client:  c1Client1
```

c. You can also run the command **mmkeyserv client show** to verify that the tenant is known to the client. The following listing shows that client `c1Client1` is registered with tenant `devG1`:

```
# mmkeyserv client show
c1Client1
        Label:              c1Client1
        Key Server:         keyserver01.gpfs.net
        Tenants:            devG1
```

d. To see the contents of the RKM stanza, run the **mmkeyserv rkm show** command. In the following listing, notice that the RKM ID of the stanza is `keyserver01_devG1`, the string that was specified in Step 5(a):

```
# mmkeyserv rkm show
keyserver01_devG1 {
type = ISKLM
kmipServerUri = tls://192.168.40.59:5696
keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
passphrase = pw4c1Client1
clientCertLabel = c1Client1
tenantName = devG1
}
```

e. You can also see the RKM stanza by displaying the contents of the `RKM.conf` file on the node:

```
# 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.c1Client1.1.p12
  passphrase = pw4c1Client1
  clientCertLabel = c1Client1
  tenantName = devG1
}
```

6. Run the **mmkeyserv key create** command to create a master encryption key in the tenant. The following command creates a master encryption key in tenant `devG1` of server `keyserver01.gpfs.net`. The command displays the UUID of the encryption key (not the key value itself) at line 3 of the listing:

```
# mmkeyserv key create --server keyserver01.gpfs.net --tenant devG1
Enter password for the key server keyserver01.gpfs.net:
KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1
```

7. Set up an encryption policy on the node.

a. Create a policy that instructs GPFS to do the encryption tasks that you want. The following policy is an example policy that instructs IBM Spectrum Scale to encrypt all files in the file system with a file encryption key and to wrap the file encryption key with a master encryption key:

```
RULE 'p1' SET POOL 'system' # one placement rule is required at all times
RULE 'Encrypt all files in filesystem with rule E1'
SET ENCRYPTION 'E1'
WHERE NAME LIKE '%'
RULE 'simpleEncRule' ENCRYPTION 'E1' IS
ALGO 'DEFAULTNISTSP800131A'
KEYS('KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1:keyserver01_devG1')
```

In the last line of the policy, the character string within single quotation marks (') is the key name. A *key name* is a compound of two parts in the following format:

```
KeyID:RkmID
```

where:

*KeyID* Specifies the UUID of the key that you created in Step 6.

*RkmID* Specifies the RKM ID that you specified in Step 5(a).

b. Run the **mmchpolicy** command to install the rule.

**CAUTION:**
**Installing a new policy with the mmchpolicy command removes all the statements in the previous policy. To add statements to an existing policy without deleting the previous contents, collect all policy statements for the file system into one file. Add the new statements to the file and install the contents of the file with mmchpolicy.**

1) Enter the following command to install the policy rules in file `enc.pol` for file system `c1FileSystem1`:

```
# mmchpolicy c1FileSystem1 /tmp/enc.pol
Validated policy `enc.pol': Parsed 3 policy rules.
Policy `enc.pol' installed and broadcast to all nodes.
```

2) You can list the new encryption policy with the following command:

```
# mmlspolicy c1FileSystem1 -L
```

8. Test the new encryption policy

a. Create a file in the file system `c1FileSystem1`:

```
# echo 'Hello World!' >/c1FilesSystem1/hw.enc
```

The policy engine detects the new file, encrypts it, and wraps the file encryption key in a master encryption key.

b. To verify that the file `hw.enc` is encrypted, enter the following command to display the encryption attribute of the file. The output shows that the file is encrypted:

```
# mmlsattr -n gpfs.Encryption /c1Filesystem1/hw.enc
file name:          /c1Filesystem1/hw.enc
gpfs.Encryption:    "EAGC????.?????????????? ??????h???????????????? ?u?~?}?????????????t??lN??
'k???*?3??C???#?)?KEY-ef07b465-cfa5-4476-9f63-544e4b3cc119?NewGlobal11?"
EncPar 'AES:256:XTS:FEK:HMACSHA512'
    type: wrapped FEK  WrpPar 'AES:KWRAP'  CmbPar 'XORHMACSHA512'
        KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1:keyserver01_devG1
```

# Simplified setup: Valid and invalid configurations

Considerable flexibility and a few restrictions govern the registering of key clients with tenants.

## Single cluster, single key server

With a single cluster and a single key server, the following rules apply:

- A single key client can register with more than one tenant.
- However, two or more key clients cannot register with the same tenant.

The following figure illustrates these rules:

- Key client `c1Client1` can register with tenants `devG1`, `devG2`, and `devG3`.
- But key client `c1Client2` cannot register with `devG1` (or `devG2` or `devG3`) because `c1Client1` is already registered there.
- Tenant `devG4` is added so that key client `c1Client2` can register with a tenant.

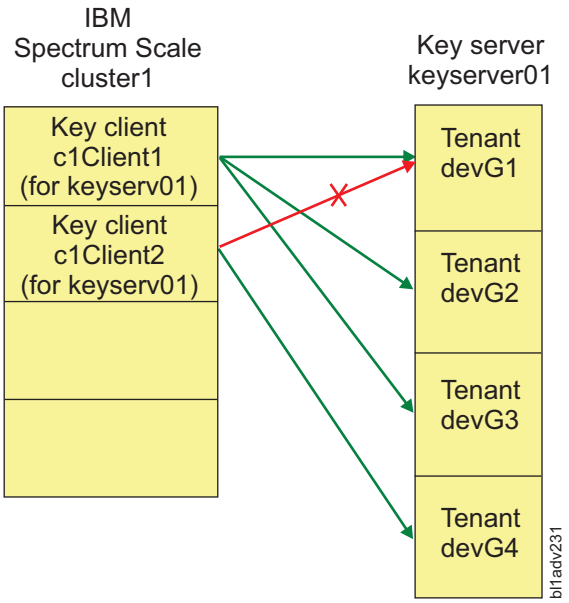



*Figure 20. Single cluster, single key server*

## Multiple clusters, single key server

With multiple clusters and a single key server, more than one key client can register with a tenant if the key clients are in different clusters.

The following figure illustrates these rules:

- With key clients `c1Client1` in `Cluster1` and `c2Client1` in `Cluster2`:
  - `c1Client1` is registered with tenants `devG1`, `devG2`, and `devG3`.
  - `c2Client1` can also register with `devG1`, `devG2`, and `devG3`, because it is in a different cluster.
- Similarly, with `c1Client2` in `Cluster1` and `c2Client1` in `Cluster2`:
  - `c1Client2` is registered with tenant `devG4`.
  - `c2Client1` can also register with `devG4`, because `c2Client1` is in a different cluster.

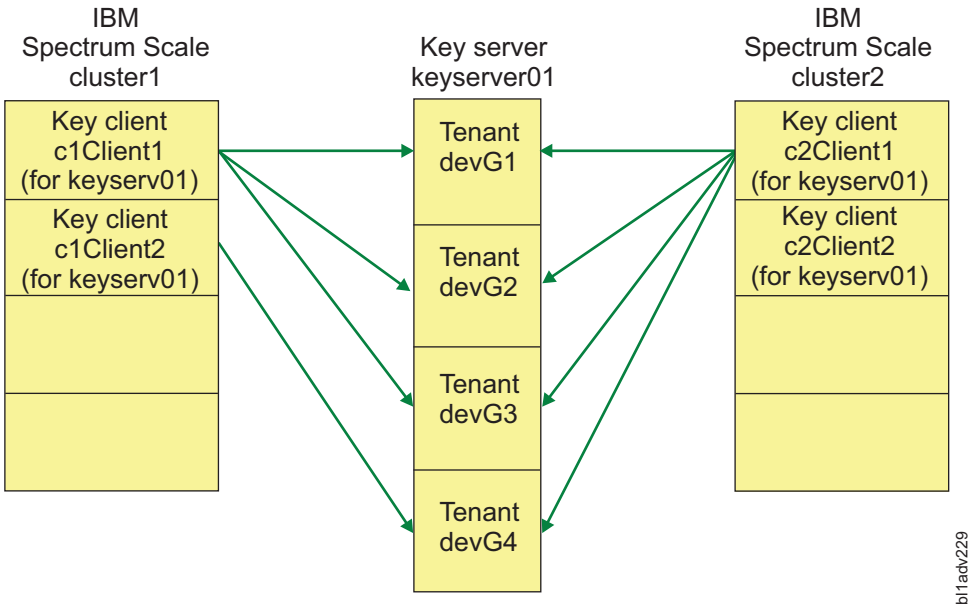



*Figure 21. Multiple clusters, single key server*

## Single cluster, multiple key servers

With a single cluster and multiple key servers, the following rules apply:

- Different key clients in the same cluster can register with different tenants in the same key server.
- But a single key client cannot register with tenants in different key servers.

The following figure illustrates these rules:

- With key clients `c1Client1` and `c1Client2`, both in `Cluster1`, it is the same situation as in Figure 20 on page 554.
  - `c1Client1` is registered with tenants `devG1`, `devG2`, and `devG3` in `keyserver01`.
  - `c1Client2` can register with tenant `devG4` in (but not with `devG1`, `devG2`, or `devG3`).
- With key client `c1Client2` in `Cluster1`:
  - `c1Client2` can register with a tenant (`devG4` in this example) in .
  - But `c1Client2` cannot also register with a tenant (`devG3`) in `keyserver02`.
- `c1Client3` was created in `Cluster1` to register with tenants `devG1` and `devG2` in `keyserver02`.

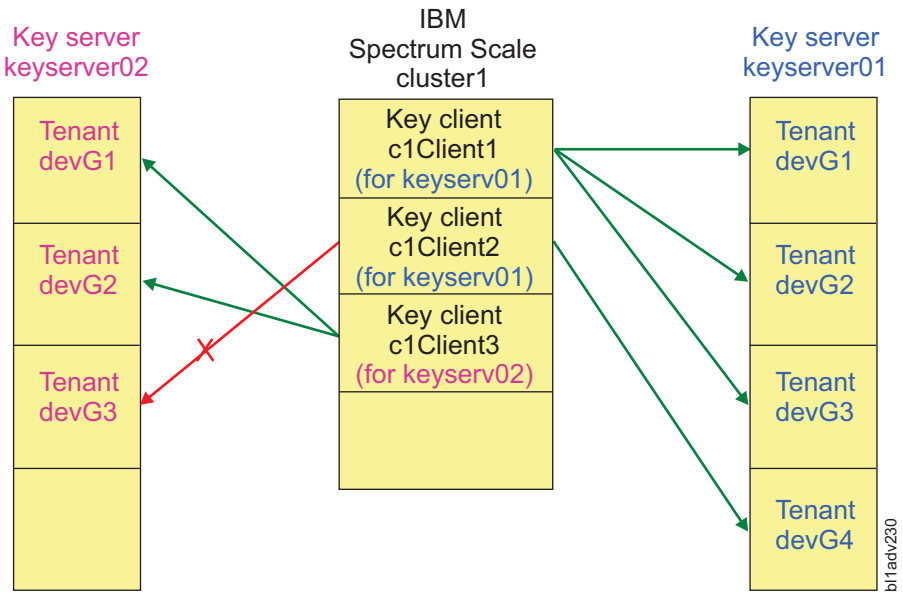



*Figure 22. Single cluster, multiple key servers*

## Simplified setup: Accessing a remote file system

See an example of how to access an encrypted file in a remote cluster.

This topic shows how to configure a cluster so that it can mount an encrypted file system that is in another cluster. In the examples in this topic, the encrypted file system is `c1FileSystem1` and its cluster is `Cluster1`. The cluster that mounts the encrypted file system is `Cluster2`.

The examples assume that `Cluster1` and `c1FileSystem1` are the cluster and file system that you configured in the topic "Configuring encryption with SKLM: Simplified setup" on page 547. You configured `Cluster1` for encryption and you created a policy that caused all the files in `c1FileSystem1` be encrypted.

To configure `Cluster2` with remote access to an encrypted file in `Cluster1`, you must configure `Cluster2` for encryption in much the same way that `Cluster1` was configured. As the following table shows, `Cluster2` must add the same key server and tenant as `Cluster1`. However, `Cluster2` must create its own key client and register it with the tenant.

**Note:** In the third column of the table, items in square brackets are connected or added during this topic. The fourth column shows the step in which each item in the third column is added.

*Table 55. Setup of Cluster1 and Cluster2*

| Item | Cluster1 | Cluster2 | Steps |
|---|---|---|---|
| File system | `c1FileSystem1` | [`c1FileSystem1_Remote`] | Step 1 |
| Connected to a key server | `keyserver01` | [`keyserver01`] | Step 2 |
| Connected to a tenant | `c1Tenant1 on keyserver01` | [`c1Tenant1 on keyserver01`] | Step 3 |
| Created a key client | `c1Client1` | [`c2Client1`] | Step 4 |
| Registered the key client to the tenant | `c1Client1 to c1Tenant1` | [`c2Client1 to c1Tenant1`] | Step 5 |
| Has access to master encryption keys | `c1Client1` | [`c2Client1`] | Step 6 |

*Table 55. Setup of Cluster1 and Cluster2 (continued)*

| Item | Cluster1 | Cluster2 | Steps |
|---|---|---|---|
| Has access to encrypted file | Local access to hw.enc in c1FileSystem1 | [Remote access to hw.enc in c1FileSystem1.] | Step 6 |

The encrypted file hw.enc is in c1FileSystem1 on Cluster1. To configure Cluster2 to have remote access to file hw.enc, follow these steps:

1. From a node in Cluster2, connect to the remote Cluster1:
   a. To set up access to the remote cluster and file system, follow the instructions in topic Chapter 21, "Accessing a remote GPFS file system," on page 281.
   b. Run the **mmremotefs add** command to make the remote file system c1FileSystem1 known to the local cluster, Cluster2:

      **Note:** c1Filesystem1_Remote is the name by which the remote file system c1FileSystem1 is known to Cluster2.

```
# mmremotefs add c1FileSystem1_Remote -f c1FileSystem1 -C Cluster1.gpfs.net -T
/c1FileSystem1_Remote -A no
mmremotefs: Propagating the cluster configuration data to all affected nodes.
    This is an asynchronous process.
Tue Mar 29 06:38:07 EDT 2016: mmcommon pushSdr_async: mmsdrfs propagation started.
```

      **Note:** After you have completed Step 1(b) and mounted the remote file system, if you try to access the contents of file hw.enc from Cluster2, the command fails because the local cluster does not have the master encryption key for the file:

```
# cat /c1FileSystem1_Remote/hw.enc
cat: hw.enc: Operation not permitted

mmfs.log:
Tue Mar 29 06:39:27.306 2016: [E]
Key 'KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1:keyserver01_devG1'
could not be fetched. The specified RKM ID does not exist;
check the RKM.conf settings.
```

2. From a node in Cluster2, connect to the same SKLM key server, keyserver01, that Cluster1 is connected to:
   a. Run the **mmkeyserv server add** to connect to keyserver01:

```
# mmkeyserv server add keyserver01
Enter password for the key 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:              7f0312e7be18efd72c9d8f37dbb832724859ba4bb5827c230e2161473e0753b367ed49d
993505bd23858541475de8e021e0930725abbd3d25b71edc8fc3de20b7c2db5cd4e865f41c7c410c1d710acf222e1c4
5189108e40568ddcbeb21094264da60a1d96711015a7951eb2655363309d790ab44ee7b26adf8385e2c210b8268c5ae
de5f82f268554a6fc22ece6efeee2a6264706e71416a0dbe8c39ceacd86054d7cc34dda4fffea4605c037d321290556
10821af85dd9819a4d7e4baa70c51addcda720d33bc9f8bbde6d292c028b2f525a0275ebea968c26f8f0c4b604719ae
3b04e71ed7a8188cd6adf68764374b29c91df3d101a941bf8b7189485ad72
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:              227b5befe89f2e55ef628da6b50db1ab842095a54e1505655e3d95fee753a7f7554868a
```

```
a79b294c503dc34562cf69c2a20128796758838968565c0812c4aedbb0543d396646a269c02bf4c5ce5acba4409a10e
ffbd47ca38ce492698e2dcdc8390b9ae3f4a47c23ee3045ff0145218668f35a63edac68201789ed0db6e5c170f5c6db
49769f0b4c9a5f208746e4342294c447793ed087fa0ac762588faf420febeb3fca411e4e725bd46476e1f9f44759a69
6573af5dbbc9553218c7083c80440f2e542bf56cc5cc18156cce05efd6c2e5fea2b886c5c1e262c10af18b13ccf38c3
533ba025b97bbe62f271545b2ab5c1f50c1dca45ce504dfcfc257362e9b43
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
```

b. Verify that the connection succeeded:

```
# mmkeyserv server show
keyserver01.gpfs.net
        Type:                ISKLM
        IPA:                 192.168.40.59
        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
```

3. From a node in `Cluster2`, add the same tenant, `c1Tenant1`, that `Cluster1` added:

a. Add the tenant `devG1`:

```
# mmkeyserv tenant add devG1 --server keyserver01
Enter password for the key server keyserver01:
mmkeyserv: [I] Tenant devG1 belongs to GPFS family exists on the key server.
Processing continues ...
mmkeyserv: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

b. Verify that the tenant is added:

```
# mmkeyserv tenant show
devG1
        Key Server:          keyserver01.gpfs.net
        Registered Client:   (none)
```

4. From a node in `Cluster2`, create a key client:

a. Create the key client `c2Client1`:

```
 # mmkeyserv client create c2Client1 --server keyserver01
Enter password for the key server keyserver01:
Create a pass phrase for keystore:
Confirm your pass phrase:
mmkeyserv: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

b. Verify that the key client is created:

```
# mmkeyserv client show
c2Client1
        Label:               c2Client1
        Key Server:          keyserver01.gpfs.net
        Tenants:             (none
```

5. From a node in `Cluster2`, register the key client to the same tenant that `Cluster1` is registered to. The RKM ID must be the same as the one that `Cluster1` uses, to allow files created with that RKM ID on `Cluster1` to be accessed from `Cluster2`. However, some of the information in the RKM stanza is different:

a. Register the client in `Cluster2` to the same tenant `c1Tenant1`:

```
# mmkeyserv client register c2Client1 --tenant devG1 --rkm-id keyserver01_devG1
Enter password for the key server :
mmkeyserv: [I] Client currently does not have access to the key.
Continue the registration process ...
mmkeyserv: Successfully accepted client certificate
mmkeyserv: Propagating the cluster configuration data to all
  affected nodes.  This is an asynchronous process.
```

b. Verify that the tenant shows that c2Client1 is registered:

```
# mmkeyserv tenant show
devG1
        Key Server:          keyserver01.gpfs.net
        Registered Client:   c2Client1
```

c. Verify that c2Client1 shows that it is registered to the c1Tenant:

```
# mmkeyserv client show
c2Client1
        Label:               c2Client1
        Key Server:          keyserver01.gpfs.net
        Tenants:             devG1
```

d. You can display the contents of the new RKM stanza:

```
 # mmkeyserv rkm show
keyserver01_devG1 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c2Client1.1.p12
  passphrase = c2Client1
  clientCertLabel = c2Client1
  tenantName = devG1
}
```

e. You can also view the RKM stanza by displaying the contents of the RKM.conf file on the command-line 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.c2Client1.1.p12
passphrase = pwc2Client1
clientCertLabel = c2Client1
tenantName = devG1
}
```

6. You can now access the encrypted file hw.enc remotely from Cluster2:

a. Verify that you can access the contents of the file hw.enc:

```
# cat /c1FileSystem1_Remote/hw.enc
Hello World!
```

b. Display the encryption attributes of the file:

```
# mmlsattr -n gpfs.Encryption /c1FileSystem1_Remote/hw.enc
file name: /c1FileSystem1_Remote/hw.enc
gpfs.Encryption: "EAGC????t!?v??????????? ???????=T???????????????? ???O?3???)??r??nV?K?OA?;????
??x,?:w?d???5)?KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1?keyserver01_devG1?"
EncPar 'AES:256:XTS:FEK:HMACSHA512'
type: wrapped FEK
WrpPar 'AES:KWRAP'
CmbPar 'XORHMACSHA512'
KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1:keyserver01_devG1
```

You can now access encrypted files on c1FileSystem1_Remote from Cluster2.

# Simplified setup: Doing other tasks

Learn how to do other tasks after you complete the simplified setup.

For the first three tasks in this topic, you need the password for your SKLM key server.

"Creating encryption keys"

"Adding a tenant"

"Managing another key server" on page 561

"Adding backup key servers" on page 565

## Creating encryption keys

This task shows how to create encryption keys in a tenant:

1. The following command creates five encryption keys in tenant `devG1` on key server `keyserver01` and displays the UUIDs of the keys on the console:

```
# mmkeyserv key create --server keyserver01.gpfs.net --tenant devG1 --count 5
Enter password for the key server keyserver01.gpfs.net:
KEY-492911c8-e3d4-4670-9868-617243d4ca57
KEY-5f24d71f-daf3-4df8-90e4-5f6475370f70
KEY-a487b01d-f092-4895-b537-139edeb57239
KEY-b449b3a2-73c5-499f-b575-fc7ba95541a8
KEY-fd3dbee9-0e6c-4662-9410-bfe3b73272b9
```

2. The following command shows the UUIDs of the encryption keys on tenant `devG1` in `keyserver01`:

```
# mmkeyserv key show --server keyserver01.gpfs.net --tenant devG1
Enter password for the key server keyserver01.gpfs.net:
KEY-492911c8-e3d4-4670-9868-617243d4ca57
KEY-5f24d71f-daf3-4df8-90e4-5f6475370f70
KEY-a487b01d-f092-4895-b537-139edeb57239
KEY-b449b3a2-73c5-499f-b575-fc7ba95541a8
KEY-d4e83148-e827-4f54-8e5b-5e1b5cc66de1
KEY-fd3dbee9-0e6c-4662-9410-bfe3b73272b9
```

   The command displays the UUIDs of the previously existing key and the five new keys.

## Adding a tenant

A tenant is a container that resides on a key server and contains encryption keys. Before a key client can request master encryption keys from a key server, you must add a tenant to the key server, create a key client, and register the key client with the tenant. For more information, see "Configuring encryption with SKLM: Simplified setup" on page 547.

In some situations, you might need to access more than one tenant on the same key server. For example, if you have several key clients that you want to use with the same key server, each key client must register with a different tenant. For more information, see "Simplified setup: Valid and invalid configurations" on page 553.

This task shows how to add a tenant, register an existing key client with the tenant, and create encryption keys in the tenant.

1. Add the tenant:
   a. Add a tenant `devG2` on `keyserver01`:

```
# mmkeyserv tenant add devG2 --server keyserver01
Enter password for the key server keyserver01:
```

   b. Verify that the tenant is added. The following command displays all the existing tenants:

```
# mmkeyserv tenant show
devG1
        Key Server:          keyserver01.gpfs.net
        Registered Client:   c1Client1
```

```
devG2
        Key Server:          keyserver01.gpfs.net
        Registered Client:   (none)
```

The tenants are devG1 and devG2.

2. Register the existing key client with the tenant:
   a. Register client c1Client1 with tenant devG2:

```
# mmkeyserv client register c1Client1 --tenant devG2 --rkm-id keyserver01_devG2
Enter password for the key server :
mmkeyserv: [I] Client currently does not have access to the key.
Continue the registration process...
mmkeyserv: Successfully accepted client certificate
```

   b. Verify that the key client is registered to the tenant:

```
# mmkeyserv client show
c1Client1
        Label:               c1Client1
        Key Server:          keyserver01.gpfs.net
        Tenants:             devG1,devG2
```

The command output shows that c1Client1 is registered to both devG1 and the new devG2.

   c. Verify the configuration of the RKM stanza. The following command displays all the RKM stanzas:

```
# mmkeyserv rkm show
keyserver01_devG1 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
  passphrase = pw_c1Client1
  clientCertLabel = label_c1Client1
  tenantName = devG1
}
keyserver01_devG2 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
  passphrase = pw_c1Client1
  clientCertLabel = label_c1Client1
  tenantName = devG2
}
```

The command shows the following relationships:

- Client c1Client1 is registered with tenant devG1 on keyserver01.
- Client c1Client1 is also registered with tenant devG2 on keyserver01.

3. Create keys in the tenant. The following command creates three keys in tenant devG2:

```
# mmkeyserv key create --server keyserver01 --tenant devG2 --count 3
Enter password for the key server keyserver01:
KEY-43cf5e69-1640-4056-b114-bdbcf2914189
KEY-4c7540cd-0346-4733-90eb-8df4c0f16008
KEY-c86a523b-e04f-4536-86a6-c6f83f845265
```

## Managing another key server

This task shows how to add a key server, add a tenant, create a new key client, and register the key client with the tenant. The steps are the same as the ones that you follow in the simplified setup:

*Table 56. Managing another key server*

| Item | Step |
|---|---|
| Install and configure SKLM. | Step 1 |
| Add a key server | Step 2 |

*Table 56. Managing another key server (continued)*

| Item | Step |
|---|---|
| Add a tenant to the key server | Step 3 |
| Create a key client | Step 4 |
| Register the key client with the tenant | Step 5 |

1. Install and configure IBM Security Key Lifecycle Manager (SKLM). For more information, see the topic "Configuring encryption with SKLM: Simplified setup" on page 547.
2. Add the key server, keyserver11. If backup key servers are available, you can add them now:
   a. Add keyserver11 and backup key servers keyserver12 and keyserver13. Enter the requested information when prompted:

```
# mmkeyserv server add keyserver11 --backup keyserver12,keyserver13
Enter password for the key server keyserver11:
The security certificate(s) from keyserver11.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:          0361e7075056
SHA-256 digest:         2a7ab79d52cca7d2cae6e88077ee48b405a9e87d03d47023fdf1d4e185f18f75
Signature:              55a4350778446ac1f74fe25016bc9efd86893b8c5e9a4c3ebc4662d7cafce8697bfbf98
f8ce62ab976fb10270a006074bd36a3c0321bb99417dcd6d9d18c06ca380f1a89aacf3d0b5d84a7fdde5d4c1b9377a0
e725d65dee819f489a9c51c2017ac6633304a3973c7e13ddc611aae6d2ba35c8571b6ca1388dbb1b91a51b00f09fe37
2846dbe0139e4f942ed317809c0b7d0cd651a3273b4df041719f99847923e5ec58517fd778d46ea44647149c5d52287
ee9705aa292c1d2942b27dd7f07d6bae2b1f29a4a818655c582ef0ce9102e70a7df68ee0c0732a66b2960959f38f964
0c599a3203ff6fcafc13f40e9922fa439d016937a00d0f5a7f571d174f277
Signature algorithm:    SHA256WithRSASignature
Key size:               2048
Issuer:                 C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=vmip131.gpfs.net
Subject:                C=US, O=IBM, OU=SKLMNode, SKLMCell, CN=vmip131.gpfs.net

Serial number:          03615d201517
SHA-256 digest:         4acb77202f885f4c6b4c858f701394f18150fd683a0d155885399bbb5b8cc0b1
Signature:              15e2011efd402b4834c677c9bcdca9914f457a9573bf1568c4d309cd1a9b873b857566c
f9653a736e34b63f8e600e1bee2450c838bbf49c6291548f0bb4ee82d8243ba60dcfbcc42f25f965fa36483441dfe7e
b2089361dbee77e333d2711ee8364f9d5005cf382a42fa90dec8f0e279b5cecb6d5ef3da2d75cdc1e70d7f4545afc13
547135c4978b717c6572b3d8c569cd44f15c0b084fe92a9e2878bcf34518882c1461e832e014d56d981ad40ef2c6760
71f49571a91e036c84ab58b3d22d0d971990624751ea6d74a420cfbf2e00d718e263184c97091404d295adb56467237
09decacebd7dbfa1927a8143bdf6d6640b72ec7c588b00cf0521c67f6efe9
Signature algorithm:    SHA256WithRSASignature
Key size:               2048
Issuer:                 C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=vmip131.gpfs.net
Subject:                C=US, O=IBM, OU=SKLMNode, SKLMCell, Root Certificate, CN=vmip131.gpfs.net

Do you trust the certificate(s) above? (yes/no) yes
```

   b. Verify that the key server is added. The following command displays information about all the existing key servers:

```
# mmkeyserv server show
keyserver01.gpfs.net
        Type:                ISKLM
        IPA:                 192.168.40.59
        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
```

```
keyserver11.gpfs.net
        Type:                ISKLM
        IPA:                 192.168.9.131
        User ID:             SKLMAdmin
        REST port:           9080
        Label:               2_keyserver11
        NIST:                on
        FIPS1402:            off
        Backup Key Servers:  keyserver12.gpfs.net,keyserver13.gpfs.net
        Distribute:          yes
        Retrieval Timeout:   120
        Retrieval Retry:     3
        Retrieval Interval:  10000
```

The command shows two key servers, `keyserver01` and the `keyserver11`.

3. Add a tenant to the key server. The name of the tenant must be unique within the same key server, but it can be the same as the name of a tenant in another key server:
   a. Add the tenant `devG1` to `keyserver11`:

```
mmkeyserv tenant add devG1 --server keyserver11
Enter password for the key server keyserver11:
```

   b. Verify that the tenant is added:

```
mmkeyserv tenant show
devG1
        Key Server:          keyserver01.gpfs.net
        Registered Client:   c1Client1

devG2
        Key Server:          keyserver01.gpfs.net
        Registered Client:   c1Client1

devG1
        Key Server:          keyserver11.gpfs.net
        Registered Client:   (none)
```

   The command shows the following tenants:

   - Tenant `devG1` on `keyserver01`.
   - Tenant `devG2` on `keyserver01`.
   - Tenant `devG1` on `keyserver11`.

4. Create a key client:

   **Note:** A key client name must be 1-16 characters in length and must be unique within an IBM Spectrum Scale cluster.

   a. Create `c1Client11` on `keyserver11`.

```
# mmkeyserv client create c1Client11 --server keyserver11
Enter password for the key server keyserver11:
Create a pass phrase for keystore:
Confirm your pass phrase:
```

   b. Verify that the client is created. The command shows all the existing key clients:

```
# mmkeyserv client show
c1Client1
        Label:               c1Client1
        Key Server:          keyserver01.gpfs.net
        Tenants:             devG1,devG2

c1Client11
        Label:               c1Client11
        Key Server:          keyserver11.gpfs.net
        Tenants:             (none)
```

The key clients are c1Client1 and c1Client11.

c. You can also display all the clients of keyserver11:

```
# mmkeyserv client show --server keyserver11
c1Client11
        Label:              c1Client11
        Key Server:         keyserver11.gpfs.net
        Tenants:            (none)
```

5. Register the key client with the tenant:

a. Verify that tenant devG1 on keyserver11 has no registered clients:

```
# mmkeyserv tenant show --server keyserver11
devG1
        Key Server:         keyserver11.gpfs.net
        Registered Client:  (none)
```

b. Register the key client c1Client11 with the devG1 on keyserver11:

```
# mmkeyserv client register c1Client11 --tenant devG1 --rkm-id keyserver11_devG1
Enter password for the key server of client c1Client11:
mmkeyserv: [I] Client currently does not have access to the key.
Continue the registration process ...
mmkeyserv: Successfully accepted client certificate
```

c. Verify that the tenant shows that the client c1Client11 is registered with it:

```
 # mmkeyserv tenant show --server keyserver11
devG1
        Key Server:         keyserver11.gpfs.net
        Registered Client:  c1Client11
```

d. You can also verify that the client shows that it is registered with tenant devG1:

```
# mmkeyserv client show --server keyserver11
c1Client11
        Label:              c1Client11
        Key Server:         keyserver11.gpfs.net
        Tenants:            devG1
```

e. Display the RKM stanzas for the cluster. They show the following relationships:

- With keyserver01, c1Client1 is registered with devG1 and devG2.
- With keyserver11, c1Client11 is registered with devG1.

```
# mmkeyserv rkm show
keyserver01_devG1 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
  passphrase = pw4c1Client1
  clientCertLabel = c1Client1
  tenantName = devG1
}
keyserver01_devG2 {
  type = ISKLM
  kmipServerUri = tls://192.168.40.59:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
  passphrase = pw4c1Client1
  clientCertLabel = c1Client1
  tenantName = devG2
}
keyserver11_devG1 {
  type = ISKLM
  kmipServerUri = tls://keyserver12.gpfs.net:5696
  kmipServerUri2 = tls://keyserver13.gpfs.net:5696
  kmipServerUri3 = tls://192.168.9.131:5696
  keyStore = /var/mmfs/ssl/keyServ/serverKmip.2_keyserver11.c1Client11.1.p12
  passphrase = pw4c1Client11
  clientCertLabel = c1Client11
  tenantName = devG1
```

f. Create encryption keys. The following command creates two keys in tenant devG1 on keyserver11.

```
# mmkeyserv key create --server keyserver11 --tenant devG1 --count 2
Enter password for the key server keyserver11:
KEY-86f601ba-0643-4f94-92b2-12c8765512cc
KEY-cdcf058f-ae30-41e8-b6f7-754e23322428
```

## Adding backup key servers

If multiple key servers exist, you can add them to an RKM stanza to provide backup capability in case the main key server becomes unavailable.

**Important:** IBM Spectrum Scale does not manage backup key servers. You must configure them and maintain them.

This task shows how to add backup key servers to the RKM stanza of one of your key clients. You can add backup key servers when you create a key server, as shown in Step 2 of the previous subtopic. Or you can add them later, as in this subtopic.

The current backup key servers for the RKM stanza are `keyserver12` and `keyserver13`. You want to add `keyserver14`, `keyserver15`, and `keyserver16` to the list.

Follow these steps:

1. Add the three key servers. You must specify the entire list of key servers, including ones that are already in the list. The following command is on one line. In the list of servers, do not put spaces on either side of the commas (,):

   ```
   # mmkeyserv rkm change keyserver11_devG1 --backup
   keyserver12,keyserver13,keyserver14,keyserver15,keyserver16
   ```

   **Attention:**

   - You can change the order in which the client tries backup key servers, by running the same command with the key servers in a different order.
   - You can delete backup key servers by specifying a list that contains the backup key servers that you want to keep and omits the ones that you want to delete.

2. To verify, show the RKM stanzas:

   ```
   # mmkeyserv rkm show
   keyserver01_devG1 {
     type = ISKLM
     kmipServerUri = tls://192.168.40.59:5696
     keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
     passphrase = pw4c1Client1
     clientCertLabel = c1Client1
     tenantName = devG1
   }

   keyserver01_devG2 {
     type = ISKLM
     kmipServerUri = tls://192.168.40.59:5696
     keyStore = /var/mmfs/ssl/keyServ/serverKmip.1_keyserver01.c1Client1.1.p12
     passphrase = pw4c1Client1
     clientCertLabel = c1Client1
     tenantName = devG2
   }

   keyserver11_devG1 {
     type = ISKLM
     kmipServerUri = tls://keyserver12.gpfs.net:5696
     kmipServerUri2 = tls://keyserver13.gpfs.net:5696
     kmipServerUri3 = tls://keyserver14.gpfs.net:5696
     kmipServerUri4 = tls://keyserver15.gpfs.net:5696
     kmipServerUri5 = tls://keyserver16.gpfs.net:5696
     kmipServerUri6 = tls://192.168.9.131:5696
     keyStore = /var/mmfs/ssl/keyServ/serverKmip.2_keyserver11.c1Client11.1.p12
   ```

```
    passphrase = pw4c1Client11
    clientCertLabel = c1Client11
    tenantName = devG1
  }
```

The command shows the following relationships:

- The configuration of `c1Client1`, `devG1`, and `keyserver01` has 0 backup servers.
- The configuration of `c1Client1`, `devG2`, and `keyserver01` has 0 backup servers.
- The configuration of `c1Client11`, `devG1`, and `keyserver11` has five backup servers.

## Configuring encryption with SKLM: Regular setup

The regular setup with IBM Security Key Lifecycle Manager (SKLM) requires IBM Spectrum Scale Advanced Edition V4.1 or later and SKLM V2.5.0.1 or later (including V2.6).

You should be aware of the following items:

The IBM Spectrum Scale node that you are configuring for encryption must have direct network access to the system where the key server is installed.

**SECURITY NOTE:** The contents of the following files are security-sensitive:

- The `RKM.conf` file:
  - For the simplified setup: `/var/mmfs/ssl/keyServ/RKM.conf`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKM.conf`
- The directory for the client keystore:
  - For the simplified setup: `/var/mmfs/ssl/keyServ`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKMcerts`

IBM Spectrum Scale reads the contents of security-sensitive files only if the following conditions are met:

- They are regular files that are owned by the root user.
- They are in the root group.
- They are readable and writable only by the user.

See the permission bits in the following examples:

- For the simplified setup:

```
-rw-------. 1 root root 2454 Mar 20 10:32 /var/mmfs/ssl/keyServ/RKM.conf
drw-------. 2 root root 4096 Mar 20 11:15 /var/mmfs/ssl/keyServ/
-rw-------. 1 root root 3988 Mar 20 11:15 /var/mmfs/ssl/keyServ/keystore_name.p12
```

  **Note:** In the simplified setup, the **mmkeyserv** command sets the permission bits automatically.

- For the regular setup and the Vormetric DSM setup:

```
-rw-------. 1 root root 2446 Mar 20 12:15 /var/mmfs/etc/RKM.conf
drw-------. 2 root root 4096 Mar 20 13:47 /var/mmfs/etc/RKMcerts
-rw-------. 1 root root 3988 Mar 20 13:47 /var/mmfs/etc/RKMcerts/keystore_name.p12
```

**CAUTION:**
**It is a good practice to take the following precautions:**

- **Ensure that the passphrase for the client certificate file is not leaked through other means, such as the shell history.**
- **Take appropriate precautions to ensure that the security-sensitive files are not lost or corrupted. IBM Spectrum Scale does not manage or replicate the files.**

**Important:** The client keystore must be record-locked when the GPFS daemon starts. If the keystore files are stored on an NFS mount, the encryption initialization process can hang. The cause is a bug that

affects the way NFS handles record locking. If you encounter this problem, upgrade your version of NFS or store your keystore file on a local file system. If an upgrade is not possible and no local file system is available, use a RAM drive to store the keystore files.

See the following subtopics for instructions:



## Part 1: Installing Security Key Lifecycle Manager

Follow the instructions in this subtopic to install and configure the IBM Security Key Lifecycle Manager (SKLM).

1. Install IBM Security Key Lifecycle Manager version 2.5.0.1 or later. For the installation, choose a system that the IBM Spectrum Scale node that you want to configure has direct network access to. For information about installing SKLM, see the IBM *Security Key Lifecycle Manager: Installation and Configuration Guide (SC27-5335)*. SKLM requires IBM WebSphere Application Server.
2. Configure SKLM to have the same FIPS 140-2 (FIPS) setting as the IBM Spectrum Scale cluster. Follow these steps:
   a. Determine the FIPS setting of the cluster by entering the following command on the command line:
      ```
      mlsconfig FIPS1402mode
      ```
      The command returns `yes` if the cluster complies with FIPS or `no` if not.
   b. On the SKLM server system, open the `SLKMConfig.properties` file.

      **Note:** The location of the `SLKMConfig.properties` file depends on the operating system:
      - On AIX, Linux, and similar operating systems:
        ```
        /opt/IBM/WebSphere/AppServer/products/sklm/config/SKLMConfig.properties
        ```
      - On Microsoft Windows:
        ```
        /opt/IBM/WebSphere/AppServer/products/sklm/config/SKLMConfig.properties
        ```
   c. Add or remove the following line from the `SKLMConfig.properties` file. Add the line to configure SKLM to comply with FIPS, or remove it to have SKLM not comply with FIPS.
      ```
      fips=on
      ```
3. Configure the SKLM server and the WebSphere Application Server to have the same NIST SP800-131a (NIST) setting as the IBM Spectrum Scale cluster. Follow these steps:

   **Note:** To configure NIST compliance in WebSphere Application Server, see the topic, "Transitioning WebSphere Application Server to the SP800-131 security standard" in the product documentation.
   a. Determine the NIST setting of the cluster by entering the following command on the command line:
      ```
      mlsconfig nistCompliance
      ```
      The command returns `SP800-131A` if the cluster complies with NIST or `off` if not.
   b. On the SKLM server system, open the `SLKMConfig.properties` file. For the location of this file, see the note in Step 2.
   c. Add the following line to configure SKLM to comply with NIST or remove it to configure SKLM not to comply with NIST:
      ```
      TransportListener.ssl.protocols=TLSv1.2
      ```

d. For all V2.5.0.x versions of SKLM, if you are configuring SKLM to comply with NIST, modify the following variable to include only cipher suites that are approved by NIST. The following statement is all on one line:

```
TransportListener.ssl.ciphersuites=TLS_RSA_WITH_AES_256_CBC_SHA256,
    TLS_RSA_WITH_AES_128_CBC_SHA256
```

4. SKLM 2.6.0.0 has a known issue that causes server certificates always to be signed with SHA1withRSA.To work around the problem, follow these steps:
   a. While the SKLM server is running, in the `SLKMConfig.properties` file, modify the `requireSHA2Signatures` property as follows:

      ```
      requireSHA2Signatures=true
      ```

   b. Do not restart the server.
   c. Generate a new server certificate and set it to be the one in use.
   d. If you restart the server, you must repeat this workaround before you can add a server certificate that is signed other than with SHA1withRSA.

## Part 2: Creating and exporting a server certificate

Follow these steps to create and export a server certificate in SKLM:

1. Create a self-signed server certificate:
   a. On the system where SKLM is running, open the graphical user interface.
   b. Click **Configuration** > **SSL/KMIP**.
   c. Click **Create self-signed certificate**.
   d. Enter the information for the certificate and click **OK**.
   e. Restart the server to verify that the server can operate with the new certificate.
2. Make a note of the label of the certificate that is in use:
   a. In the SKLM graphical user interface, click **Advanced Configuration** > **Server Certificates**.
   b. Select the certificate that is identified as being in use. Click **Modify** and make a note of the certificate label. You need it in Step 3.
3. Export the certificate through the command-line interface. Follow these steps:
   a. On the SKLM server system, open a command-line window.
   b. Change to the *WAS_HOME*/bin directory. The location of this directory depends on the operating system:
      - On AIX, Linux, and similar operating systems:

        ```
        /opt/IBM/WebSphere/AppServer/bin
        ```

      - On Microsoft Windows:

        ```
        drive:\Program Files (x86)\IBM\WebSphere\AppServer\bin
        ```

   c. Enter the following command to start the command-line interface to SKLM:
      - On AIX, Linux, and similar operating systems:

        ```
        ./wsadmin.sh -username SKLMAdmin -password mypwd -lang jython
        ```

      - On Microsoft Windows:

        ```
        wsadmin -username SKLMAdmin -password mypwd -lang jython
        ```

   d. In the SKLM command line interface, enter the following command:

      ```
      print AdminTask.tklmCertList('[-alias labelSSCert]')
      ```

      where:

      *labelSSCert*
      : Specifies the certificate label of the self-signed server certificate. You made a note of the label in Step 2.

      SKLM responds with output like the following example:

```
CTGKM0001I Command succeeded.
uuid = CERTIFICATE-7005029a-831d-405f-af30-4bf0177909de
alias = server
key store name = defaultKeyStore
key state = ACTIVE
issuer name = CN=server
subject name = CN=server
creation date = 13/03/2014 16:27:13 Eastern Daylight Time
expiration date = 09/03/2015 07:12:30 Eastern Daylight Time
serial number = 1394363550
```

e. Make a note of the UUID of the certificate that is displayed on line 2 of the output. You need it in Part 3.

f. To export the certificate, from the SKLM command line interface, enter the following command on one line:

```
print AdminTask.tklmCertExport('[-uuid certUUID -format base64 -fileName fileName]')
```

where:

*certUUID*
Specifies the UUID that you made a note of in the previous step.

*fileName*
Specifies the path and file name of the certificate file in which the server certificate is stored.

SKLM exports the self-signed server certificate into the specified file.

g. Close the SKLM command line interface.

h. Copy the certificate file to a temporary directory on the GPFS node that you are configuring for encryption.

4. In SKLM, create a device group and keys for the IBM Spectrum Scale cluster:

a. In the SKLM graphical user interface, click **Advanced Configuration** > **Device Group**.

b. In the Device Group table, click **Create**.

c. In the Create Device Group window, follow these steps:

1) Select the **GPFS** device family.

2) Enter an appropriate name, such as `GPFS_TENANT1`. The name is case-sensitive.

**Note:** A *tenant* in the **mmkeyserv** command equates to a device group in SKLM.

3) Make a note of the name. You need it in Part 3 when you create an RKM stanza.

4) Complete any other fields and click **Create**.

d. After SKLM creates the device group, it prompts you to add devices and keys. Do not add any devices or keys. Instead, click **Close**. Keys are created in the next step.

5. Create keys for the device group.

a. In the SKLM graphical user interface, in the Key and Device Management table, select the device group that you created in Step 4. In these instructions the device group is named GPFS_TENANT1.

b. Click **Go to** > **Manage keys and services**.

c. In the management page for GPFS_TENANT1, click **Add** > **Key**.

d. Enter the following information:

- The number of keys to be created
- The three-letter prefix for key names. The key names are internal SKLM names and are not used for GPFS encryption.

e. Make a note of the UUID of the key, such as `KEY-326a1906-be46-4983-a63e-29f005fb3a15`. You need it in Part 3.

## Part 3: Configuring the remote key management (RKM) back end

To configure a remote key management (RKM) back end, you must create a client keystore and an RKM stanza in the `RKM.conf` file on the IBM Spectrum Scale node.

1. Create and configure a client keystore. Follow these steps:
   a. On the IBM Spectrum Scale node that you are configuring for encryption, create the following subdirectory to contain the client keystore:

      ```
      /var/mmfs/etc/RKMcerts
      ```

   b. The following command creates the client keystore, stores a private key and a client certificate in it, and also stores the trusted SKLM server certificate into it. From the command line, enter the following command on one line:

      ```
      mmauth gencert --cname clientName --cert serverCertFile --out /var/mmfs/etc/RKMcerts/SKLM.p12
      --label clientCertLabel --pwd-file passwordFile
      ```

      where the parameters are as follows:

      **--cname** *clientName*
      The name of the client that is used in the certificate.

      **--cert** *serverCertFile*
      The path and file name of the file that contains the SKLM server certificate. You extracted this certificate from SKLM and copied the certificate file to the node in Part 2, Step 3h.

      **--out** */var/mmfs/etc/RKMcerts/SKLM.p12*
      The path and file name of the client keystore.

      **--label** *clientCertLabel*
      The label of the client certificate in the keystore. The label can be 1-20 characters in length.

      **--pwd-file** *passwordFile*
      The path of a text file that contains the password for the client keystore. The password can be 1-20 characters in length.

   **Important:** The new keystore must be record-locked when the GPFS daemon starts. If the keystore files are stored on an NFS mount, the encryption initialization process can hang. The cause is a bug that affects the way NFS handles record locking. If you encounter this problem, upgrade your version of NFS or store your keystore file on a local file system. If an upgrade is not possible and no local file system is available, use a RAM drive to store the keystore files.

2. Create an `RKM.conf` file and add a stanza to it that contains the information that is necessary to connect to the SKLM key server. The `RKM.conf` file must contain a stanza for each connection between a key client, an SKLM device group, and a key server.
   a. In a text editor, create a new text file with the following path and name:

      ```
      /var/mmfs/etc/RKM.conf
      ```

   b. Add a stanza with the following format:

      ```
      stanzaName {
         type = ISKLM
         kmipServerUri = tls://raclette.zurich.ibm.com:5696
         keyStore = /var/mmfs/etc/RKMcerts/SKLM.p12
         passphrase = a_password
         clientCertLabel = a_label
         tenantName = GPFS_Tenant0001
      }
      ```

      where the rows of the stanza have the following meaning:

      **stanzaName**
      A name (RKM ID) for the stanza. Make a note of the name: you need it in the next step.

It is a good practice to use a format like the following one to ensure that the RKM ID is unique:

```
keyServerName_tenantName
```

where *tenantName* is the name that you provide in the last line of stanza. For example, the RKM ID for the key server and key client in these instructions is: `raclette_GPFS_Tenant0001`.

**type** Always ISKLM.

**kmipServerUri**
The DNS name or IP address of the SKLM server and the KMIP SSL port. You can find this information on the main page of the SKLM graphic user interface. The default port is 5696.

You can have multiple instances of this line, where each instance represents a different backup key server. The following example has the primary key server and two backup key servers:

```
stanzaName {
   type = ISKLM
   kmipServerUri = tls://raclette.zurich.ibm.com:5696
   kmipServerUri = tls://raclette.fondue.ibm.com:5696
   kmipServerUri = tls://raclette.fondue2.ibm.com:5696
   keyStore = /var/mmfs/etc/RKMcerts/SKLM.p12
   passphrase = a_password
   clientCertLabel = a_label
   tenantName = GPFS_Tenant0001
}
```

If the GPFS daemon cannot get an encryption key from the primary key server, it tries the backup key servers in order.

**keyStore**
The path and name of the client keystore. You specified this parameter in Step 1.

**passphrase**
The password of the client keystore and client certificate. You specified this parameter in Step 1.

**clientCertLabel**
The label of the client certificate in the client keystore. You specified this parameter in Step 1.

**tenantName**
The name of the SKLM device group. See "Part 1: Installing Security Key Lifecycle Manager" on page 567.

3. Set up an encryption policy on the node that you are configuring for encryption.
   a. Create a policy that instructs GPFS to do the encryption tasks that you want. The following policy is an example policy. It instructs IBM Spectrum Scale to encrypt all files in the file system with a file encryption key (FEK) and to wrap the FEK with a master encryption key (MEK):

```
RULE 'p1' SET POOL 'system' # one placement rule is required at all times
RULE 'Encrypt all files in filesystem with rule E1'
SET ENCRYPTION 'E1'
WHERE NAME LIKE '%'
RULE 'simpleEncRule' ENCRYPTION 'E1' IS
ALGO 'DEFAULTNISTSP800131A'
KEYS('KEY-326a1906-be46-4983-a63e-29f005fb3a15:SKLM_srv')
```

In the last line, the character string within single quotation marks (') is the key name. A *key name* is a compound of two parts in the following format:

```
KeyID:RkmID
```

where:

**KeyID**
Specifies the UUID of the key that you created in the SKLM graphic user interface in Part 2.

**RkmID**
Specifies the name of the RKM backend stanza that you created in the `/var/mmfs/etc/RKM.conf` file.

b. Install the policy rule with the **mmchpolicy** command.

**CAUTION:**
**Installing a new policy with the mmchpolicycommand removes all the statements in the previous policy. To add statements to an existing policy without deleting the previous contents, collect all policy statements for the file system into one file. Add the new statements to the file and install the contents of the file with the mmchpolicy command.**

4. Import the client certificate into the SKLM server:
   a. On the IBM Spectrum Scale node that you are configuring for compression, send a KMIP request to SKLM. To send a KMIP request, try to create an encrypted file on the node. The attempt fails, but it causes SKLM to put the client certificate in a list of pending certificates in the SKLM key server. The attempt fails because SKLM does not yet trust the client certificate. See the following example:

```
# touch /gpfs0/test
touch: cannot touch `/gpfs0/test': Permission denied
# tail -n 2 /var/adm/ras/mmfs.log.latest
Thu Mar 20 14:00:55.029 2014: [E] Unable to open encrypted file: inode 46088,
Fileset fs1, File System gpfs0.
Thu Mar 20 14:00:55.030 2014: [E] Error: key
'KEY-326a1906-be46-4983-a63e-29f005fb3a15:SKLM_srv' could not be fetched (RKM
reported error -1004).
```

   b. In the graphical user interface of SKLM, on the main page, click **Pending client device communication certificates.**
   c. Find the client certificate in the list and click **View**.
   d. Carefully check that the certificate that you are importing matches the one created in the previous step, then click **Accept and Trust**.
   e. On the resulting screen, provide a name for the certificate and click **Accept and Trust** again.
   f. On the node that you are configuring for compression, try to create an encrypted file as you did in Step (a). This time the command succeeds. Enter an **mmlsattr** command to list the encryption attributes of the new file:

```
# touch /gpfs0/test
# mmlsattr -n gpfs.Encryption /gpfs0/test
file name: /gpfs0/test
gpfs.Encryption: "EAGC????f????????????????? ??????w?^??>???????????? ?L4??
_-???V}f???X????,?G?<sH??0?)??M?????)?KEY-326a1906-be46-4983-a63e-29f005fb3a15?
sklmsrv?)?KEY-6aaa3451-6a0c-4f2e-9f30-d443ff2ac7db?RKMKMIP3?"
EncPar 'AES:256:XTS:FEK:HMACSHA512'
type: wrapped FEK WrpPar 'AES:KWRAP' CmbPar 'XORHMACSHA512'
KEY-326a1906-be46-4983-a63e-29f005fb3a15:sklmsrv
```

From now on, the encryption policy rule causes each newly created file to be encrypted with a file encryption key (FEK) that is wrapped in a master encryption key (MEK). You created the key in a device group in the SKLM server and included its UUID as part of a key name in the security rule.

**Important:** See the security note and the caution at the beginning of this topic, before Part 1.

## Part 4: Enabling encryption on other nodes

1. To replicate an encryption configuration on another node, copy the contents of the `/var/mmfs/etc` directory to the node.

2. To create a different encryption configuration on another node, follow the steps that are described in the preceding subtopics. Note the following design points:
   - On a single node
     - The `RKM.conf` file can contain multiple stanzas. Each stanza represents a connection between a key client and an SKLM device group.
     - You can create multiple keystores.
   - Across different nodes
     - The contents of `RKM.conf` files can be different.
     - The contents of keystores can be different.
     - If an encryption policy succeeds on one node and fails on another in the same cluster, verify that the failing node has the correct client keystore and stanza.

## Configuring encryption with the Vormetric DSM key server

Setting up an encryption environment with Vormetric Data Security Manager (DSM) key server requires IBM Spectrum Scale Advanced Edition V4.2.1 or later and Vormetric DSM V5.2.3 or later.

You should be aware of the following items:

The IBM Spectrum Scale node that you are configuring for encryption must have direct network access to the system where the key server is installed.

**SECURITY NOTE:** The contents of the following files are security-sensitive:

- The `RKM.conf` file:
  - For the simplified setup: `/var/mmfs/ssl/keyServ/RKM.conf`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKM.conf`
- The directory for the client keystore:
  - For the simplified setup: `/var/mmfs/ssl/keyServ`
  - For the regular setup and the Vormetric DSM setup: `/var/mmfs/etc/RKMcerts`

IBM Spectrum Scale reads the contents of security-sensitive files only if the following conditions are met:

- They are regular files that are owned by the root user.
- They are in the root group.
- They are readable and writable only by the user.

See the permission bits in the following examples:

- For the simplified setup:

```
-rw-------. 1 root root 2454 Mar 20 10:32 /var/mmfs/ssl/keyServ/RKM.conf
drw-------. 2 root root 4096 Mar 20 11:15 /var/mmfs/ssl/keyServ/
-rw-------. 1 root root 3988 Mar 20 11:15 /var/mmfs/ssl/keyServ/keystore_name.p12
```

  **Note:** In the simplified setup, the **mmkeyserv** command sets the permission bits automatically.

- For the regular setup and the Vormetric DSM setup:

```
-rw-------. 1 root root 2446 Mar 20 12:15 /var/mmfs/etc/RKM.conf
drw-------. 2 root root 4096 Mar 20 13:47 /var/mmfs/etc/RKMcerts
-rw-------. 1 root root 3988 Mar 20 13:47 /var/mmfs/etc/RKMcerts/keystore_name.p12
```

**CAUTION:**
**It is a good practice to take the following precautions:**

- **Ensure that the passphrase for the client certificate file is not leaked through other means, such as the shell history.**
- **Take appropriate precautions to ensure that the security-sensitive files are not lost or corrupted. IBM Spectrum Scale does not manage or replicate the files.**

**Important:** The client keystore must be record-locked when the GPFS daemon starts. If the keystore files are stored on an NFS mount, the encryption initialization process can hang. The cause is a bug that affects the way NFS handles record locking. If you encounter this problem, upgrade your version of NFS or store your keystore file on a local file system. If an upgrade is not possible and no local file system is available, use a RAM drive to store the keystore files.

See the following subtopics for instructions:

"Part 1: Creating credentials for the key client"
"Part 2: Configuring the Vormetric DSM key server" on page 577
"Part 3: Configuring the IBM Spectrum Scale node" on page 578

## Part 1: Creating credentials for the key client

- Some of the commands in the following instructions require you to specify values for the following two parameters:

  **--fips** Specifies whether the key client complies with the requirements of FIPS 140-2.

  **--nist** Specifies whether security transport for the key client complies with the NIST SP800-131A recommendations.

  For both parameters, follow these guidelines:

  – If the key client complies, set the parameter to **on**; otherwise, set the parameter to **off**.

  – Specify the same setting for each parameter as the setting in the IBM Spectrum Scale cluster. To display these settings, enter the following two commands:

```
mmlsconfig nistCompliance
mmlsconfig FIPS1402mode
```

Follow these steps:

1. On the IBM Spectrum Scale node that you are configuring for encryption, run the **mmgskkm** command to create the client credentials. Enter the following command on one line:

```
/usr/lpp/mmfs/bin/mmgskkm gen --prefix prefix cname cname --pwd pwd --fips fips --nist nist
    --days valid_days --keylen keylen
```

   where:

   **--prefix** *prefix*
   Specifies the path and file name prefix of the directory where the output files are generated. For example, if you want directory `/var/mmfs/etc/RKMcerts` to contain the output files, and you want the output files to have the prefix `kcVormetric`, you can specify the parameter as follows:

```
--prefix /var/mmfs/etc/RKMcerts/kcVormetric
```

   **--cname** *cname*
   Specifies the name of the IBM Spectrum Scale key client. Valid characters are alphanumeric characters, hyphen (-), and period (.). The name can be up to 54 characters long. In Vormetric DSM, names are not case-sensitive, so the use of uppercase letters is not recommended. For more information, see the Vormetric DSM documentation.

**--pwd** *pwd*
Specifies the password for the private key that this command creates.

**--fips** *fips*
Specifies whether the key client complies with FIPS 140-2. Specify **on** or **off**.

**--nist** *nist*
Specifies whether the key client complies with NIST SP800-131a. Specify **on** or **off**.

**--validdays** *validdays*
Specifies the number of days that the client certificate is valid.

**--keylen** *keylen*
Specifies the length in bits of the RSA key that is generated.

In the following example, the current directory is the output directory. Enter the command on one line:

```
/usr/lpp/mmfs/bin/mmgskkm gen --prefix kcVormetric --cname kcVormetric --pwd pwpkVormetric
    --fips off --nist on --days 180 --keylen 2048
```

The output files are a client certificate, a private key, and a public key. For example:

```
kcVormetric.cert
kcVormetric.priv
kcVormetric.pub
```

2. Run the **mmgskkm** command again to create a PKCS#12 keystore and to store the certificate and private key of the client in it. Enter the following command on one line:

```
/usr/lpp/mmfs/bin/mmgskkm store --cert certFile --priv privFile --label label --pwd pwd --out keystore
```

where:

**--cert** *certFile*
Specifies the client certificate file that you created in Step 1.

**--priv** *privFile*
Species the private key file that you created in Step 1.

**--label** *label*
Specifies the label under which the private key is stored in the keystore.

**--pwd** *pwd*
Specifies the password of the keystore. You can use the same password that you specified for the private key in Step 1.

**--out** *keystore*
The file name of the keystore.

In the following example, the current directory contains the client credentials from Step 1. The command is entered on one line:

```
mmgskkm store --cert kcVormetric.cert --priv kcVormetric.priv --label lapkVormetric
    --pwd pwpkVormetric --out ksVormetric.keystore
```

The output file is a keystore that contains the client credentials of the key client:

```
ksVormetric.keystore
```

**Important:** The keystore must be record-locked when the GPFS daemon starts. If the keystore files are stored on an NFS mount, the encryption initialization process can hang. The cause is a bug that affects the way NFS handles record locking. If you encounter this problem, upgrade your version of NFS or store your keystore file on a local file system. If an upgrade is not possible and no local file system is available, use a RAM drive to store the keystore files.

3. Retrieve the certificate chain of the Vormetric DSM server.

**Note:** Before you can do this next step, you must install the Vormetric DSM server, set up the DSM networking configuration, and set up the server certificate. If you do not, then you might not be able to connect to the DSM server or you might retrieve an invalid, default certificate chain.
Enter the following command on one line:

```
/usr/lpp/mmfs/bin/mmsklmconfig restcert --host host --port port --prefix prefix --keystore pwdfile
    --keypass keypass --fips fips --nist nist
```

where:

**--host** *host*
: Specifies the name or IP address of the remote system where the Vormetric DSM server is running.

**--port** *port*
: Specifies the port on the remote system for communicating with the Vormetric DSM server.

**--prefix** *prefix*
: Specifies the path and file name prefix of the directory where the files in the certificate chain are stored. For example, if you want to store the certificate chain in the directory `/var/mmfs/etc/RKMcerts`, and you want the certificate files to have the prefix `DSMServer`, you can specify the parameter as follows:

```
--prefix /var/mmfs/etc/RKMcerts/DSMServer
```

**--keystore** *keystore*
: Specifies the path and file name of the client keystore that you created in Step 2.

**--keypass** *pwdFile*
: Specifies a text file that contains the password of the client keystore as the first line. You must create this text file. Store the password that you provided in Step 2.

**--fips** *fips*
: Specifies whether the key client complies with FIPS 140-2. Specify **on** or **off**.

**--nist** *nist*
: Specifies whether the key client complies with NIST SP800-131a. Specify **on** or **off**.

In the following example, the current directory contains the client keystore that was created in Step 2. Enter the command on one line:

```
/usr/lpp/mmfs/bin/mmssklmconfig restcert --host hostVormetric --port 443 --prefix DSM
     --keystore ksVormetric.keystore --keypass pwdFile --fips off --nist on
```

The command connects to the DSM server, retrieves the server certificate chain, and stores each certificate into a separate local file in Base64-encoded DER format. Each file name has the format *prefixN*.**cert**, where *prefix* is the prefix that you specified in the command and *N* is a digit that begins at 0 and increases by 1 for each certificate in the chain, as in the following example:

DSM0.cert

DSM1.cert

DSM2.cert

4. Verify that the SHA-256 fingerprint in each retrieved certificate matches the fingerprint of the DSM server:
   a. To display the details of each certificate, enter the following sequence at the client command line, where *prefix* is the prefix that you provided in Step 3:

      ```
      for c in prefix*.cert; do /usr/lpp/mmfs/bin/mmgskkm print --cert $c; done
      ```

   b. Log in to the graphical user interface of the DSM server and display its SHA-256 fingerprint.
   c. Verify that the fingerprints in the certificates match the fingerprint in the DSM server.
5. Add the certificates to the PKCS#12 keystore of the key client as trusted certificates. Enter the following command on one line:

   ```
   /usr/lpp/mmfs/bin/mmgskkm trust --prefix prefix --pwd pwd --out keystore --label labelServer
       --fips fips --nist nist
   ```

where:

**--prefix** *prefix*
Specifies the prefix that you specified in Step 3.

**--pwd** *pwd*
Specifies the password of the client keystore, which you provided in Step 3.

**--out** *keystore*
Specifies the path name of the keystore of the key client.

**--label** *serverLabel*
Specifies the label under which the server certificate chain is stored in the client keystore.

**--fips** *fips*
Specifies whether the key client complies with FIPS 140-2. Specify **on** or **off**.

**--nist** *nist*
Specifies whether the key client complies with NIST SP800-131a. Specify **on** or **off**.

In the following example, the current directory contains the client keystore and the certificate chain. Enter the following command on one line:

```
/usr/lpp/mmfs/bin/mmssklmconfig trust --prefix DSM --pwd pwpkVormetric --out ksVormetric.keystore
    --label laccVormetric --fips off --nist on
```

The keystore of the key client contains the following items:

- Client credentials
- The certificate chain of the Vormetric DSM key server as trusted certificates

## Part 2: Configuring the Vormetric DSM key server

The following instructions describe how to configure the Vormetric Data Security Manager (DSM) key server to communicate with an IBM Spectrum Scale key client.

In DSM, a *host* is a system to which DSM provides security services. In these instructions, the host is the IBM Spectrum Scale node that you are configuring for encryption. A DSM *domain* is an administrative group of one or more hosts. In these instructions, the domain contains the single IBM Spectrum Scale node. For more complex configurations, see the DSM product documentation.

1. Install a Key Management Interoperability Protocol (KMIP)-enabled license in DSM.

   **Important:** You must complete this step before you create a DSM domain. For security reasons, you cannot create a KMIP-enabled domain in DSM until you install a KMIP-enabled license. For example, you cannot create a regular domain, install a KMIP-enabled license, and then convert the domain to a KMIP-enabled domain.

   a. On the DSM Management Console, click **System** > **License**.
   b. Select a KMIP-enabled license that you obtained from DSM.
   c. Click **Upload License File**.

   The license is installed.
2. Create a DSM domain.
   a. On the DSM Management Console, click **Domains** > **Manage Domains**.
   b. Follow the instructions to create a domain. Make sure that you configure the domain as **KMIP Supported**.
3. Create a Domain and Security Administrator for the new domain.

   **Note:** In these instructions, a single Domain and Security Administrator is created who combines the responsibilities of administering the domain and controlling its security. For security reasons, you might want to create a Domain Administrator and a Security Administrator as separate roles. For more information, see the DSM documentation.

a. Log in as the DSM System Administrator. On the Management Console, click **Administrators**.
b. On the Administrators page, click **Add**.
c. In the Add Administrator window, complete all the input fields except the **RSA User ID** field. In the **User Type** field, click **Domain and Security Administrator**.

   **Note:** The passwords are temporary. The new administrator must enter a new password on the first login to the DSM Management Console.
d. Click **OK**.
e. Limit the scope of the administrator's control to the domain that you created in Step 2.

4. Add a host to the domain.
   a. Log in as the new administrator:
      1) Enter a password when prompted.
      2) Select **I am a local domain administrator**.
      3) Enter or select the domain name from Step 2.
   b. On the Management Console, click **Hosts** > **Hosts**.
   c. On the Hosts screen, click **Add** to add a KMIP host. Set the **Host Name** to the name that you specified for the key client (the value for the **cname** parameter) when you created the client credentials in Part 1. In these instructions, the key client name is `kcVormetric`.
   d. In the list of hosts, select the host that you created in the previous step. Click **Import KMIP Cert**. If no **Import KMIP Cert** button is displayed, verify that the DSM license is KMIP-enabled and that you created the domain after you installed the KMIP-enabled license.
   e. In the window that opens, go through the directories of the IBM Spectrum Scale node to the directory that contains the client certificate file. Select the certificate file.
5. Create one or more keys for the client to use as master encryption keys (MEKs).
   a. From the DSM Management Console, click **Keys** > **Key Templates.** Follow the instructions to create a key template. Select **AES256** as the key algorithm.
   b. Create a key from the template. Specify a name for the key and then select the template.
   c. Make a note of the UUID of the key. You need it in Part 3.

## Part 3: Configuring the IBM Spectrum Scale node

1. Create an `RKM.conf` file and add a remote key management (RKM) stanza to it that contains the information that is necessary to communicate with the Vormetric DSM key server.
   a. On the IBM Spectrum Scale node, create a text file with the following path and name:
      ```
      /var/mmfs/etc/RKM.conf
      ```
   b. Add a stanza with the following format:
      ```
      stanzaName {
         type = KMIP
         kmipServerUri = tls://raclette.zurich.ibm.com:5696
         keyStore = /var/mmfs/etc/RKMcerts/ksVormetricDMS.p12
         passphrase = a_password
         clientCertLabel = a_label
      }
      ```

      where the rows of the stanza have the following meaning:

      **stanzaName**
      A name (RKM ID) for the stanza. Make a note of the name: you need it in the next step.

      It is a good practice to use a format like the following one to ensure that the RKM ID is unique:

      *keyServerName_keyClientName*

where *keyClientName* is the key client name from Part 1, Step 1. For example, the RKM ID for the key server and key client in these instructions is: `raclette_kcVormetric`.

**type** Always KMIP for the Vormetric DSM server.

**kmipServerUri**
The DNS name or IP address of the DSM server and the DSM SSL port.

**keyStore**
The path and name of the client keystore from Part 1.

**passphrase**
The password of the client keystore and client certificate from Part 1.

**clientCertLabel**
The label of the client certificate in the client keystore from Part 1.

2. Set up an encryption policy on the node that you are configuring for encryption.
   a. Create a policy that instructs GPFS to do the encryption tasks that you want. The following policy is an example policy. It instructs IBM Spectrum Scale to encrypt all files in the file system with a file encryption key (FEK) and to wrap the FEK with a master encryption key (MEK):

```
RULE 'p1' SET POOL 'system' # one placement rule is required at all times
RULE 'Encrypt all files in filesystem with rule E1'
SET ENCRYPTION 'E1'
WHERE NAME LIKE '%'
RULE 'simpleEncRule' ENCRYPTION 'E1' IS
ALGO 'DEFAULTNISTSP800131A'
KEYS('01-10:raclette_kcVormetric')
```

   In the last line, the character string within single quotation marks (') is the key name. A *key name* is a compound of two parts in the following format:

```
KeyID:RkmID
```

   where:

   **KeyID**
   Specifies the UUID of the master encryption key that you created in the DSM Management Console in Part 2.

   **RkmID**
   Specifies the name of the RKM stanza that you created in the `/var/mmfs/etc/RKM.conf` file in Step 1.

   b. Install the policy rule with the **mmchpolicy** command.

   **CAUTION:**
   **Installing a new policy with the mmchpolicy command removes all the statements in the previous policy. To add statements to an existing policy without deleting the previous contents, collect all policy statements for the file system into one file. Add the new statements to the file and install the contents of the file with the mmchpolicy command.**

   From now on, the encryption policy rule causes each newly created file to be encrypted with a file encryption key (FEK) that is wrapped in a master encryption key (MEK).

## Secure deletion

Secure deletion refers to both erasing files from the file system and erasing the MEKs that wrapped the FEKs that were used to encrypt the files.

### Securely deleting files in a fileset

After files have been removed from a fileset using standard file system operations (such as **unlink** and **rm**), the tenant administrator might decide to securely delete them. For example, suppose that until that

point, the FEKs of all files in the fileset were encrypted with the MEK with key name `KEY-old:isklmsrv`. To cause the secure deletion of all removed files, the administrator must perform the following steps:

1. Create a new MEK and note its key name (in this example, `KEY-new:isklmsrv`).
2. Modify the appropriate encryption policy **KEYS** statement in the encryption policy to encrypt new files with the new MEK (for example, `KEY-new:isklmsrv`) instead of the old one (`KEY-old:isklmsrv`).
3. Create and apply a migration (rewrapping) policy (**CHANGE ENCRYPTION KEYS**) to scan all files, unwrap the wrapped FEK entries of files that have been wrapped with the old key (`KEY-old:isklmsrv`), and rewrap them with the new key (`KEY-new:isklmsrv`); this step ensures that the FEKs of existing files will be accessible in the future.
4. Remove the old key, `KEY-old:isklmsrv`. This step commits the secure deletion of all files that were previously unlinked (and whose FEKs had therefore not been rewrapped with the new MEK, `KEY-new:isklmsrv`).
5. On each node that has ever done I/O to a file encrypted with the old key (`KEY-old:isklmsrv`), run the following command:

   ```
   /usr/lpp/mmfs/bin/tsctl encKeyCachePurge 'KEY-old:isklmsrv'
   ```

From this point on, the new key will be used for encryption, which will be performed transparently to the application.

**Note:** The **mmdelfs** command will *not* perform any secure deletion of the files in the file system to be deleted. **mmdelfs** only removes all the structures for the specified file system. To securely delete files, you need to perform the following steps:

1. Identify all MEKs currently used to wrap the FEKs of files in the file system to be deleted. If this information is not available through other means, obtain it by doing the following:
   a. Invoke **mmlsattr -n gpfs.Encryption** on all files of the file system.
   b. Parse the resulting output to extract all the distinct key names of the MEKs that are used.

   **Note:** These are the possible ways that an MEK might be in use in a file system:
   a. The MEK is, or was at some point, specified in an encryption rule in the policy set on the file system.
   b. An FEK rewrap has been run, rewrapping an FEK with another MEK.
2. Determine whether the identified MEKs were used to wrap FEKs in other file systems.

   **WARNING:** If the same MEKs were used to wrap FEKs in other file systems, deleting those MEKs will result in irreparable data loss in the other file systems where those MEKs are used. Before deleting such MEKs from the key servers, you must create one or more new MEKs and rewrap the files in the other file systems.
3. After appropriately handling any MEKs that were used to wrap FEKs in other file systems (as explained in the warning), delete the identified MEKs from their RKMs.

## Secure deletion and encryption key cache purging

The key servers that store the MEKs know how to manage and securely delete keys. After an MEK is gone, all files whose FEKs were encrypted with that MEK are no longer accessible. Even if the data blocks corresponding to the deleted files are retrieved, the contents of the file can no longer be reconstructed, since the data cannot be decrypted.

However, if the MEKs have been cached for performance reasons (so that they do not have to be fetched from the server each time a file is created or accessed), the MEKs must also be purged from the cache to complete the secure deletion.

You can use the following command to purge a given key from the key cache, or to clean the entire cache, of an individual node:

```
/usr/lpp/mmfs/bin/tsctl encKeyCachePurge {Key | all}
```

where:

*Key*
is the key ID, specified with the *KeyId*:*RkmId* syntax.

**all**
specifies that the entire key cache is to be cleaned.

The scope of this command is limited to the local node and must be run on all nodes that have accessed the MEKs you are purging in order to ensure secure deletion.

## Encryption and FIPS compliance

The **FIPS1402mode** configuration variable controls whether the use of crypto-based security mechanisms (if they are to be used at all, per the GPFS administrator) is to be provided by software modules that are certified according to the requirements and standards described by the Federal Information Processing Standards (FIPS) 140 Publication Series. When in FIPS 140-2 mode, GPFS uses the FIPS 140-2 approved cryptographic provider(s); IBMJCEFIPS (certificate 376) and/or IBMJSSEFIPS (certificate 409) and/or IBM Crypto for C (ICC) (certificate 384) for cryptography. The certificates are listed on the NIST website.

The value of **FIPS1402mode** can be changed with the **mmchconfig** command. The default value for this variable is **no**. With **FIPS1402mode=no**, Linux nodes will use kernel encryption modules for direct I/O. If a cluster is configured with **FIPS1402mode=yes**, Linux nodes whose kernels are not running in FIPS mode will see a performance degradation when using direct I/O. The GPFS daemon on the node must be restarted in order for the new setting to take place.

**Note:** In IBM Spectrum Scale V4.2.0 and earlier, in a Power 8, little-endian environment, the setting **FIPS1402mode=no** is required for the following operations:

- File encryption
- Secure communications between nodes. For more information, see the following descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:
  - **-l** *CipherList* parameter of the **mmauth** command
  - **cipherList** parameter of the **mmchconfig** command
- CCR enablement. For more information, see the following descriptions in the *IBM Spectrum Scale: Command and Programming Reference*:
  - **--ccr-enable** parameter of the **mmchcluster** command
  - **--ccr-enable** parameter of the **mmcrcluster** command.

## Encryption and NIST compliance

Encryption always uses NIST-compliant mechanisms.

## Encryption in a multicluster environment

If an encrypted file system is made available via remote mounts, then the remote cluster also requires network reachability to the key server. In some deployments, the key servers might be located at the home cluster, but in others one might choose to locate key servers in a high-availability configuration on both home and remote clusters. The order of the servers can be specified in the set of RKM back ends such that the server closest to the local node is accessed first.

## Encryption in a Disaster Recovery environment

While setting up multiple key servers in a high-availability configuration is important to ensure that the MEKs remain available, it is especially important in a Disaster Recovery environment. It is a good practice to place at least one key server on each site to ensure that keys remain available if access to an entire site is lost. For more information, see "Identifying multiple RKM back ends in a high-availability configuration" on page 546.

## Encryption and backup/restore

GPFS will deliver all data to **mmbackup** and other external backup solutions in **cleartext** whether or not the data is encrypted in GPFS. Any backups that are taken **will not** preserve the encryption status or the encrypted content of the data. Files that are recreated upon restore will be considered for encryption status based on the policy in place on the file system at the time of the restore operation.

## Encryption and snapshots

GPFS preserves the encryption status of files when they are copied into global or fileset snapshots. Global snapshot restore will restore files precisely as they are in the snapshot, including FEKs and MEKs. For details about how fileset snapshot restore functions, see the description of the **mmrestorefs** command in the *IBM Spectrum Scale: Command and Programming Reference*.

# Chapter 34. Managing certificates to secure communications between GUI web server and web browsers

The IBM Spectrum Scale system supports self-signed and trusted certificates that are provided by a certificate authority (CA) to secure communications between the system and web browser.

During system setup, an initial self-signed certificate is created to use for secure connections between the GUI web servers and web browsers. Based on the security requirements for your system, you can create either a new self-signed certificate or install a signed certificate that is created by certify authority. Self-signed certificates can generate web browser security warnings and might not comply with organizational security guidelines. The self-signed certificates are stored in the Liberty profile SSL keystore, which is located at the `resources/security` directory in the server. You can find this directory in the following path: `/opt/ibm/wlp/usr/servers/gpfsgui/resources/security`.

The trusted certificates are created by a third-party certificate authority. These certificate authorities ensure that certificates have the required security level for an organization based on purchase agreements. Trusted certificates usually have higher security controls for encryption of data and do not cause browser security warnings. Trusted certificates are also stored in the Liberty profile SSL keystore.

Major web browsers trust the CA-certified certificates by default and therefore they can confirm that the certificate received by the GUI server can be trusted. You can either buy a signed certificate from a trusted third-party authority or create you own certificate and get it certified. You can use both self-signed and trusted certificates. However, using a trusted is the preferred way because the browser trusts this certificate automatically without any manual interventions.

## Obtaining and importing a signed-certificate from a trusted certificate authority

You need to perform the following steps to obtain and import a signed-certificate from a trusted certificate authority:

1. Generate a private key by issuing the following command:

   ```
   openssl genrsa -out <nameOfYourKey>.key 2048
   ```

2. Generate the certificate request as shown in the following example:

   ```
   openssl req -new -key <nameOfYourKey>.key -out <nameOfYourKey>.csr
   ```

   The system prompts you to enter the following details:

   ```
   Country Name (2 letter code) [XX]:
   State or Province Name (full name) []:
   Locality Name (eg, city) [Default City]:
   Organization Name (eg, company) [Default Company Ltd]:
   Organizational Unit Name (eg, section) []:
   Common Name (eg, your name or your server's hostname) []:
   Email Address []:
   Please enter the following 'extra' attributes to be sent with your certificate request
   A challenge password []:
   An optional company name []:
   ```

3. Send the certificate request to a trusted certificate authority to get a certificate file.
4. Create a PKCS12 store containing the certificate as shown in the following example:

   ```
   openssl pkcs12 -export -in <yourCertificateFile> -inkey <nameOfYourKey>.key
   <nameOfYourPKCS12File>.p12
   ```

   The system prompts to set the export password as shown in the following example:

   ```
   Enter export Password: <yourPasswrod>
   Verifying - Enter export Password: <yourPasswrod>
   ```

5. Generate a Java keystore file (.jks) by using the keytool. It is stored in the following directory: /opt/ibm/wlp/java/jre/bin. Ensure that you set the paths to the Java keystore file properly in the system. Issue the following commands to generate a Java keystore file.

   ```
   <PathToKeytool>/keytool -importkeystore -srckeystore
   <NameOfYourPKCS12File>.p12 -destkeystore
   <NameOfYourJKSFile>.jks -srcstoretype pkcs12
   ```

   The system prompts you to enter the destination keystore password. You need to use the same password that you used while creating the PKCS12 store.

   ```
   Enter destination keystore password: <yourPassword>
   Re-enter new password: <yourPassword>
   Enter source keystore password: <yourPassword>
   ```

6. Copy the new Java keystore file to the directory named security, which is stored in the GUI server. This directory is located at the following location in the GUI server: /opt/ibm/wlp/usr/servers/gpfsgui/resources/security. It is the default place where keystore files are stored.

   ```
   cp <NameOfYourJKSFile>.jks <pathToSecurityDir>
   ```

7. You need to define the password to access the Java Keystone file in the server.xml file, which is stored in the GUI server. If you want to encode your password so that it does not get stored in plain text, use a security utility, which is stored in the following directory: /opt/ibm/wlp/bin. The supported encodings are XOR and AES.

   ```
   <PathToSecurityUtility>/securityUtility encode --encoding=<xor or aes> <yourPassword>
   ```

8. Edit keystore entry of the server.xml file for the GUI server to enable the new key. The server.xml file is located at the following location: /opt/ibm/wlp/usr/servers/gpfsgui. The following entry is on one line:

   ```
   <keyStore id="defaultKeyStore" password="<yourPassword> or <yourEncodedPassword>"
    location="<nameOfYourJKSFile>.jks" />
   ```

9. Restart the IBM Spectrum Scale management GUI by issuing the following commands:

   **For RHEL7 or SLES12:** systemctl restart gpfsgui
   **For RHEL6:** service gpfsgui restart

# Chapter 35. Securing protocol data

The data cannot be secured only by authenticating and authorizing the users to access the data. You also need to ensure that the communication channel that is used to raise authentication requests and data transfer is secured. The security features associated with the protocols that you use to store and access data also help to provide data in transit security for the protocol data.

The secured data access by clients through protocols is achieved through the following two steps:

1. Establishing secured connection between the IBM Spectrum Scale system and the authentication server.

   When the client raises an authentication request to access the data, the IBM Spectrum Scale system interacts with the external authentication servers like Active Directory or LDAP based on the authentication configuration. You can configure the security services like TLS and Kerberos with the external authentication server to secure the communication channel between the IBM Spectrum Scale system and the external authentication server.
2. Securing the data transfer.

   The actual data access wherein the data transfer is made secured with the security features that are available with the protocol that you use to access the data.

The following diagram depicts the data in transit security implementation in the IBM Spectrum Scale system.

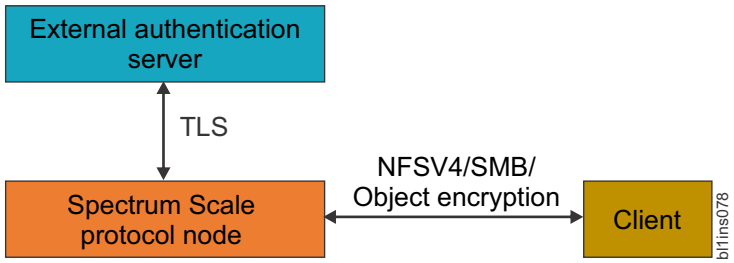



*Figure 23. Implementation of data in transit security for protocol data*

## Secured connection between the IBM Spectrum Scale system and the authentication server

You can configure the following authentication servers to configure file and object access:

- Microsoft Active Directory (AD)
- Lightweight Directory Access Protocol (LDAP)
- Keystone

AD and LDAP can be used as the authentication server for both file and object access. Configuring the Keystone server is a mandatory requirement for the object access to function. The keystone needs to interact with the authentication server to resolve the authentication requests. You can configure either an internal or external keystone server for object access. The following table lists the security features that are used to secure the corresponding authentication server.

*Table 57. Security features that are used to secure authentication server.*

| Authentication server | Supported protocols | Security features |
|---|---|---|
| Active Directory | File and Object | Kerberos for file and TLS for object. |

*Table 57. Security features that are used to secure authentication server (continued).*

| Authentication server | Supported protocols | Security features |
|---|---|---|
| LDAP | File and Object | Both TLS and Kerberos for file and only TLS for object. |
| Keystone | Object | SSL certificate to enable HTTPS connection |

## Secured data transfer

The secured data transfer over the network is based on the security features available with the protocols that are used to access the data.

### Secured SMB data transfer

SMB protocol version 3 and later has the following capabilities to provide tighter security for the data transfers:

1. Secured dialect negotiation
2. Improved signing
3. Secured transmission

The dialect negotiation is used to identify the highest level dialect both server and client can support. The system administrator can enable SMB encryption by using the `smb encrypt` setting at the export level. The following three modes are available for the secured SMB access:

- Automatic
- Mandatory
- Disabled

When the SMB services are enabled, the SMB encryption is enabled in the automatic mode by default.

**Note:** SMB supports per-export encryption, which allows the administrators to selectively enable or disable encryption per SMB share.

### Secured NFS data transfer

The following security methods are used with NFSV4 protocol:

1. Enabling squashing

   Any file requests that are made by the root user on the client system is considered as a potential threat. By default, root user requests are treated as if it is made by the user nobody on the server. If you disable squashing, the root user on the client gets the same level of access to files on the system as the root user on the server. You can disable squashing if, for example, you want to run an administrative task on the client system that has the exported directories that are stored on it.
2. Using Kerberos

   Kerberos is a network authentication protocol that ensures secure communication over a network. You can use Kerberos instead of local UNIX UIDs and GIDs to authenticate users. Kerberos can operate in the following modes to provide improved security:

   - **Kerberos v5:** Authentication only
   - **Kerberos v5 with integrity:** Authentication and data integrity
   - **Kerberos v5 with privacy:** Authentication and encryption of data traffic between the client and the server. Most secure, but it might cause some performance issues because of the heavy processing required for encryption.
3. Enabling port security

You can enable or disable the port security in all communications between the client and the server. When port security is enabled, the system does not allow access to the requests that originate from ports where the port number is greater than the hardcoded threshold value of 1024.

**Note:** The NFS security features can be configured per NFS export by the system administrator based on the requirement.

**Secured object data access**

The IBM Spectrum Scale system provides access to the Object Storage with the help of OpenStack Keystone Identity Service. The Keystone server that is provided by IBM Spectrum Scale is recommended to be used only for IBM Spectrum Scale Object workload.

For secure communication between the clients and the IBM Spectrum Scale Object, the system administrator needs to configure HAProxy for SSL termination, traffic encryption, and load balancing of the requests to IBM Spectrum Scale Object. The HAProxy needs to be set up on an external system that is not a part of the IBM Spectrum Scale cluster. For more information on how to configure HAProxy, see the documentation of the corresponding Linux distribution that you selected.

# Planning for protocol data security

It is recommended to adhere to the following best practices when you plan to set up data security:

- When Windows clients use SMB 3.0, always configure SMB share with `smb encrypt = mandatory`.
- Avoid clients who used SMB 2.1 from accessing SMB data.
- Always enforce to use UNC with NetBIOS name of the cluster to access SMB data.
- To ensure NFSV4 encryption, use an authentication method that is configured with Kerberos.

# Configuring protocol data security

The data security features associated with protocols facilitate to configure a secured way for the clients to raise the data access request and to transfer data from the IBM Spectrum Scale system to the client system.

# Enabling secured connection between the IBM Spectrum Scale system and authentication server

You need to secure the communication channel between the IBM Spectrum Scale system and authentication server to secure the authentication server and hence to prevent unauthorized access to data and other system resources.

## Securing AD server

To secure the AD server that is used for file access, configure it with Kerberos and to secure AD used for object access, configure it with TLS.

In the AD-based authentication for file access, Kerberos is configured by default. The following steps provide an example on how to configure TLS with AD, while it is used for object access.

1. Ensure that the CA certificate for AD server is placed under `/tmp` directory with the name *ldap_cacert.pem*; specifically, on the protocol node where the command is run. Perform validation of CA cert availability with desired name at required location as shown in the following example:

```
# stat /tmp/ldap_cacert.pem
File: ⌂/tmp/ldap_cacert.pem⌂
Size: 2130 Blocks: 8 IO Block: 4096 regular file
Device: fd00h/64768d Inode: 103169903 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 0/ root) Gid: ( 0/ root)
```

```
Context: unconfined_u:object_r:user_tmp_t:s0
Access: 2015-01-23 12:37:34.088837381 +0530
Modify: 2015-01-23 12:16:24.438837381 +0530
Change: 2015-01-23 12:16:24.438837381 +0530
```

2. To configure AD with TLS authentication for object access, issue the **mmuserauth service create** command:

```
# mmuserauth service create --type ad --data-access-method object
--user-name "cn=Administrator,cn=Users,dc=IBM,dc=local"
--password "myPassword" --base-dn "dc=IBM,DC=local"
--enable-server-tls --ks-dns-name myKeystoneDnsName
--ks-admin-user admin --servers myADserver
--user-id-attrib cn --user-name-attrib sAMAccountName
--user-objectclass organizationalPerson --user-dn "cn=Users,dc=IBM,dc=local"
--ks-swift-user swift --ks-swift-pwd myKWSwiftPassword
Object configuration with LDAP (Active Directory) as identity
backend is completed successfully.
Object Authentication configuration completed successfully.
```

**Note:** The value that you specify for `--servers` must match the value in the TLS certificate; otherwise the command fails.

3. To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
FILE access not configured
PARAMETERS               VALUES
-----------------------------------------------

OBJECT access configuration: AD
PARAMETERS               VALUES
-----------------------------------------------
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        true
ENABLE_KS_SSL            false
USER_NAME                cn=Administrator,cn=Users,dc=IBM,dc=local
SERVERS                  myADserver
BASE_DN                  dc=IBM,DC=local
USER_DN                  cn=users,dc=ibm,dc=local
USER_OBJECTCLASS         organizationalPerson
USER_NAME_ATTRIB         sAMAccountName
USER_ID_ATTRIB           cn
USER_MAIL_ATTRIB         mail
USER_FILTER              none
ENABLE_KS_CASIGNING      false
KS_ADMIN_USER            admin
```

## Securing LDAP server

To secure the LDAP server that is used for file access, configure it with TLS and Kerberos and to secure LDAP server that is used for object access, configure it with TLS.

Provides examples of how to configure LDAP with TLS and Kerberos to secure the LDAP server when it is used for file and object access.

1. To configure LDAP with TLS and Kerberos as the authentication method for file access, issue the **mmuserauth service create** command as shown in the following example:

```
# 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 --enable-kerberos
--kerberos-server es-pune-host-01 --kerberos-realm example.com
```

The system displays the following output:

```
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 the following output:

```
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          es-pune-host-01
KERBEROS_REALM           example.com

OBJECT access not configured
PARAMETERS               VALUES
-------------------------------------------------
```

2. To configure LDAP with TLS as the authentication method for object access, issue the **mmuserauth service create** command as shown in the following example:

```
# mmuserauth service create --type ldap --data-access-method object
--user-name "cn=manager,dc=essldapdomain" --password "Passw0rd"
--base-dn dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com --enable-server-tls
--ks-dns-name c40bbc2xn3 --ks-admin-user mamdouh --servers 192.0.2.11
--user-dn "ou=People,dc=essldapdomain" --ks-swift-user swift
--ks-swift-pwd Passw0rd
```

The system displays the following output:

```
Object configuration with LDAP as identity backend is completed successfully.
Object Authentication configuration completed successfully.
```

To verify the authentication configuration, use the **mmuserauth service list** command as shown in the following example:

```
# mmuserauth service list
```

The system displays the following output:

```
FILE access not configured
PARAMETERS               VALUES
-------------------------------------------------

OBJECT access configuration : LDAP
PARAMETERS               VALUES
-------------------------------------------------
ENABLE_ANONYMOUS_BIND    false
ENABLE_SERVER_TLS        true
ENABLE_KS_SSL            false
USER_NAME                cn=manager,dc=essldapdomain
SERVERS                  192.0.2.11
BASE_DN                  dc=isst,dc=aus,dc=stglabs,dc=ibm,dc=com
USER_DN                  ou=people,dc=essldapdomain
USER_OBJECTCLASS         posixAccount
USER_NAME_ATTRIB         cn
```

```
USER_ID_ATTRIB          uid
USER_MAIL_ATTRIB        mail
USER_FILTER             none
ENABLE_KS_CASIGNING     false
KS_ADMIN_USER           mamdouh
```

### Securing Keystone server

The Keystone server that is used by the IBM Spectrum Scale system supports SSL. The SSL certificate provides secure communication while resolving the authentication requests. When Keystone is configured with authentication servers such as LDAP or AD, the system can be configured to establish a secured communication between AD or LDAP and Keystone by using TLS encryption. For more information on configuring AD or LDAP-based authentication with TLS, see the **mmuserauth service create** command. The IBM Spectrum Scale for Object Storage can also be configured with an external Keystone server. If the external Keystone server contains SSL certificate in place, then the system administrator can configure secured communication with the IBM Spectrum Scale system by following some manual steps.

Provides an example on how to configure secured object access.

1. Ensure that the following certificates are placed at the following location in the IBM Spectrum Scale system:
   - `certfile = /etc/keystone/ssl/certs/signing_cert.pem`
   - `keyfile = /etc/keystone/ssl/private/signing_key.pem`
   - `ca_certs = /etc/keystone/ssl/certs/signing_cacert.pem`

   To provide customized signing keys, follow these steps:

   a. Put the following three keys in the `/tmp` folder:
      - `/tmp/signing_cacert.pem`
      - `/tmp/signing_cert.pem`
      - `/tmp/signing_key.pem`

   b. Specify the **--enable-ks-casigning** parameter when you run the **mmuserauth** command in Step 2.
2. Issue the **mmuserauth service create** command as shown in the following example to configure local authentication method for object.

```
# mmuserauth service create --type local --data-access-method object
--ks-dns-name c40bbc2xn3 --ks-admin-user admin --ks-admin-pwd Passw0rd
--enable-ks-casigning
/usr/lpp/mmfs/bin/mmcesobjcrbase: Configuring Keystone server in
/gpfs/cesfs/ces/object/keystone
Object configuration with local (Database) as identity backend
is completed successfully.
Object Authentication configuration completed successfully.
```

3. Issue the **mmuserauth service list** command to verify the authentication configuration.

```
# mmuserauth service list
FILE access not configured
PARAMETERS              VALUES
-------------------------------------------------

OBJECT access configuration : LOCAL
PARAMETERS              VALUES
-------------------------------------------------
ENABLE_KS_SSL           false
ENABLE_KS_CASIGNING     true
KS_ADMIN_USER           admin
```

## Securing data transfer

The data in transit security is configured by using the security features that are available with the protocol that is used for data I/O.

## Securing NFS data transfer

Securing the NFS data transfer over the network is achieved by using the Kerberos-based encryption that is available with NFSV4 protocol. You can use Kerberos to encrypt the data that is transferred over the network and also to secure the communication with the authentication server.

The following example shows how to enable data security to ensure secured NFS data transfer.

1. 1. Create a keytab file for protocol nodes in IBM Spectrum Scale cluster. To create keytab file, you need to create a principal `nfs/<node-fqdn>` for each protocol node. Issue the following commands on the system that hosts the KDC server. In the following example, the sample commands are submitted on the Linux system that hosts MIT KDC server:

   ```
   $ addprinc -randkey nfs/<protocol-node1-fqdn>
   $ addprinc -randkey nfs/<protocol-node2-fqdn>

   .....

   &mldr; $ addprinc -randkey nfs/<protocol-nodeN-fqdn>
   $ ktadd -k /tmp/krb5.keytab nfs/<protocol-node1-fqdn>
   $ ktadd -k /tmp/krb5.keytab nfs/<protocol-node2-fqdn>

   .....

   &mldr; $ ktadd -k /tmp/krb5.keytab nfs/<protocol-nodeN-fqdn>
   ```

2. Ensure that the keytab file that is created is placed under the `/tmp` directory as *krb5.keytab*; specifically, on the node where the IBM Spectrum Scale authentication commands are submitted. Perform validation of keytab file availability with the required name and location:

   ```
   # stat /tmp/krb5.keytab
   File: ⌂/tmp/krb5.keytab⌂
   Size: 502 Blocks: 8 IO Block: 4096 regular file
   Device: fd00h/64768d Inode: 103169898 Links: 1
   Access: (0600/-rw-------) Uid: ( 0/ root) Gid: ( 0/ root)
   Context: unconfined_u:object_r:user_tmp_t:s0
   Access: 2015-01-23 14:31:18.244837381 +0530
   Modify: 2015-01-23 12:45:05.475837381 +0530
   Change: 2015-01-23 12:45:05.476837381 +0530
   Birth: -
   ```

3. Issue the **mmuserauth service create** command on the IBM Spectrum Scale protocol node as shown in the following example:

   ```
   #  mmuserauth service create --data-access-method file --type ldap
   --servers 192.0.2.17 --base-dn dc=example,dc=com
   --user-name "cn=manager,dc=example,dc=com" --password secret --enable-kerberos
   --kerberos-server 192.0.2.17 --kerberos-realm example.com --netbios-name cktest
   File Authentication configuration completed successfully.
   ```

4. Issue the **mmuserauth service list** command to see the current authentication configuration as shown in the following example:

   ```
   # mmuserauth service list
   ILE 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                 9.118.46.17
   NETBIOS_NAME            cktest
   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          9.118.46.17
KERBEROS_REALM           example.com

OBJECT access not configured
PARAMETERS               VALUES
-----------------------------------------------
```

5. Create Kerberos exports with krb5, krb5i, and krb5p security features on the IBM Spectrum Scale node.

```
# mmcrfileset gpfs0 krb5
Fileset krb5 created with id 2 root inode 47898.

# mmlinkfileset gpfs0 krb5 -J /ibm/gpfs0/krb5
Fileset krb5 linked at /ibm/gpfs0/krb5

# mmnfs export add /ibm/gpfs0/krb5 --client
    "*(ACCESS_TYPE=RW,SQUASH=no_root_squash,SECTYPE=krb5p)"
The NFS export was created successfully.

# mmcrfileset gpfs0 krb5i
Fileset krb5i created with id 3 root inode 47900.

# mmlinkfileset gpfs0 krb5i -J /ibm/gpfs0/krb5i
Fileset krb5i linked at /ibm/gpfs0/krb5i

# mmnfs export add /ibm/gpfs0/krb5i --client
    "*(ACCESS_TYPE=RW,SQUASH=no_root_squash,SECTYPE=krb5p)"
The NFS export was created successfully.

# mmcrfileset gpfs0 krb5p
Fileset krb5p created with id 4 root inode 47895.

# mmlinkfileset gpfs0 krb5p -J /ibm/gpfs0/krb5p
Fileset krb5p linked at /ibm/gpfs0/krb5p

# mmnfs export add /ibm/gpfs0/krb5p --client
    "*(ACCESS_TYPE=RW,SQUASH=no_root_squash,SECTYPE=krb5p)"
The NFS export was created successfully.

# mmnfs export list
```

The system displays output similar to this:

```
Path               Delegations    Clients
-------------------------------------------
/ibm/gpfs0/krb5    none           *
/ibm/gpfs0/krb5i   none           *
/ibm/gpfs0/krb5p   none           *
/ibm/gpfs0/nfsexp1 none           *
```

6. Issue the **mmnfs export list** command with krb5 option to see the authentication only configuration.

```
# mmnfs export list --nfsdefs /ibm/gpfs0/krb5
```

The system displays output similar to this:

```
Path            Delegations  Clients Access_Type Protocols Transports Squash        Anonymous_uid Anonymous_gid SecType PrivilegedPort Export_id DefaultDelegation Manage_Gids NFS_Commit
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
/ibm/gpfs0/krb5    none          *       RW         3,4       TCP       NO_ROOT_SQUASH -2           -2           KRB5    FALSE         2         none              FALSE       FALSE
```

7. Issue the **mmnfs export list** command with krb5i option to see the authentication and data integrity configuration.

```
# mmnfs export list --nfsdefs /ibm/gpfs0/krb5i
```

The system displays output similar to this:

```
Path             Delegations    Clients Access_Type Protocols Transports Squash        Anonymous_uid Anonymous_gid SecType PrivilegedPort Export_id DefaultDelegation Manage_Gids NFS_Commit
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
/ibm/gpfs0/krb5i   none          *       RW         3,4       TCP       NO_ROOT_SQUASH -2           -2           KRB5I   FALSE         3         none              FALSE       FALSE
```

8. Issue the **mmnfs export list** command with krb5p option to see the authentication and privacy configuration.

   ```
   # mmnfs export list --nfsdefs /ibm/gpfs0/krb5p
   ```

   The system displays output similar to this:

```
Path              Delegations    Clients Access_Type Protocols Transports Squash        Anonymous_uid Anonymous_gid SecType PrivilegedPort Export_id DefaultDelegation Manage_Gids NFS_Commit
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
/ibm/gpfs0/krb5p  none           *       RW          3,4       TCP        NO_ROOT_SQUASH -2            -2            KRB5P   FALSE          4         none              FALSE       FALSE
```

## Securing SMB data transfer

Secured SMB data transfer can be enabled when you are using SMB3 and later.

You can either enable or disable encryption of the data in transit by using the **mmsmb export create** command as shown in the following example:

```
# mmsmb export create secured_export  /ibm/gpfs0/secured_export --option "smb encrypt=mandatory"
```

## Secured object data transfer

For secure communication between the clients and the IBM Spectrum Scale Object, the system administrator needs to configure HAProxy for SSL termination, traffic encryption, and load balancing of the requests to IBM Spectrum Scale Object. The HAProxy needs to be set up on an external system that is not a part of the IBM Spectrum Scale cluster. For more information on how to configure HAProxy, see the documentation of the corresponding Linux distribution that you selected.

## Data security limitations

The following are the protocol data security limitations:

- The file system encryption feature is not supported for securing file and object protocol data.
- The SMB encryption is available only on SMB3 and later. All the limitations that are identified by Microsoft also apply to SMB encryption. There are no SMB encryption limitations that are specific to IBM Spectrum Scale.
- Delegations cannot be used with NFS in the Kerberos environment, because they cause the NFSV4 server to crash. If you use NFS in the Kerberos environment, you should disable delegations.

# Chapter 36. Transparent Cloud Tiering

This topic provides a brief description about managing your cloud storage tiers by using IBM Spectrum Scale.

## Administering the Transparent Cloud Tiering service

This topic provides a brief description on how to manage a Transparent Cloud Tiering service in IBM Spectrum Scale.

## Starting the Transparent Cloud Tiering services

This topic describes how to start Transparent Cloud Tiering service in IBM Spectrum Scale.

Before you try to start the Transparent Cloud Tiering service on a CES node, ensure that the node is designated as the Transparent Cloud Tiering node. For more information, see "Designating the Transparent Cloud Tiering nodes" on page 41.

Start the Transparent Cloud Tiering service before you run any of the Transparent Cloud Tiering commands.

To start the Transparent Cloud Tiering service, issue a command according to this syntax:

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

For example, to start the service on all Transparent Cloud Tiering nodes as provided in the node class, *TCTNodeClass1*, issue this command:

```
mmcloudgateway service start TCTNodeClass1
```

If you provide this command without any arguments, the service is started on the current node.

**Note:** You must run this command on any one node that is designated as the Transparent Cloud Tiering node. The service is automatically started on all other nodes.

**Next step**: "Designating the Transparent Cloud Tiering nodes" on page 41.

## Stopping the Transparent Cloud Tiering service

This topic describes the procedure for stopping the Transparent Cloud Tiering service.

To stop the Transparent Cloud Tiering service on a specific node or a list of nodes, issue a command according to this syntax:

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

For example, to stop the service on the node, 10.11.12.13, issue this command:

```
mmcloudgateway service stop -N 10.11.12.13
```

You must run this command on the Transparent Cloud Tiering node.

**Note:** Before you stop the Transparent Cloud Tiering service, ensure that no migration or recall operation is running on the system where the service is stopped.

## Testing a cloud storage tier

After you create a cloud storage tier by using the **mmcloudgateway account create** command, you must test it to ensure that a connection is established between your file system and the cloud storage tier.

To test the connection, issue a command according to this syntax:

```
mmcloudgateway account test --cloud-nodeclass TCTNodeClass1 --cloud-name CloudName
```

where,

- TCTNodeClass1 is the node class defined for the Transparent Cloud Tiering nodes.
- CloudName is the name that you specify for the cloud account.

For example, to test the cloud storage tier, tctenew, created on IBM Cloud Object Storage cloud type, issue this command:

```
mmcloudgateway account test --cloud-nodeclass TCTNodeClass1 --cloud-name tctnew
```

The system displays output similar to this:

```
Cloud Status : Configured cloud account is Active!
mmcloudgateway: Command completed.
```

command. For more information on the usage of the command, see **mmcloudgateway** command in the *IBM Spectrum Scale: Command and Programming Reference*.

## Listing cloud storage tiers

This topic describes the procedure for listing the cloud storage tiers that are configured in the IBM Spectrum Scale cluster.

To list the configured cloud storage providers within Transparent Cloud Tiering, issue this command:

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

where,

- *CloudNodeClass* is the node class that is associated with the file system.
- *CloudName* is the unique name that is provided while configuring a cloud storage tier.

For example,

```
# mmcloudgateway account list --cloud-name mcstore

Configured Cloud Details:
Cloud Provider Name : mcstore
Cloud Provider Tenant Id : admin
Cloud Provider URL : http://9.114.96.186:5000/v2.0
Cloud Provider Type : swift-keystone
Cloud Provider User Name : admin
Cloud Provider Enabled : true
Filesystem Root Path : /gpfs
Container : mcstore-8610578645082165495
```

**Note:** Use this option to view more details about the configured cloud provider.

## Administering files

This topic provides a brief description on administering files on the cloud storage tier by using Transparent Cloud Tiering.

## Applying a policy on a Transparent Cloud Tiering node

This topic provides description with an example about creating an ILM policy and then applying this policy to a Transparent Cloud Tiering node.

After a cloud account is configured, you can apply an ILM policy file to configure a cloud storage tier. The policy configuration is done by using IBM Spectrum Scale standard ILM policy query language statements.

For more information on ILM policies, see Chapter 22, "Information lifecycle management for IBM Spectrum Scale," on page 293 .

You must create a policy and then apply this policy on the Gateway node for the ILM-based migration and recall to work for the cloud storage tier.

**Note:** Administrators must consider appropriate high and low disk utilization threshold values that are applicable in the data center environment.

A sample policy rule and the steps to apply the policy on a node are as follows:

```
 /* Sample policy.rules file for using Gateway functionality */
 /* Define an external pool for the off-line storage */
 define(
 exclude_list,
  (
   FALSE
   OR PATH_NAME LIKE '%/.mcstore/%'
  )
 )
define(
access_age,
(DAYS(CURRENT_TIMESTAMP) - DAYS(ACCESS_TIME))
)
define(
mb_allocated,
(INTEGER(KB_ALLOCATED / 1024))
)
define(
weight_expression,
(CASE
 /*=== The file is very young, the ranking is very low ===*/
 WHEN access_age <= 1 THEN 0
 /*=== The file is very small, the ranking is low ===*/
 WHEN mb_allocated < 1 THEN access_age
 /*=== The file is resident and large and old enough,
 the ranking is standard ===*/
 ELSE mb_allocated * access_age
 END)
 )
/* Define an external pool for the off-line storage */
RULE EXTERNAL POOL 'mcstore' EXEC '/opt/ibm/MCStore/bin/mcstore' OPTS '-F'
/* Define migration rule with a threshold to trigger low space events
and move data to the external off-line pool. When on-line usage
exceeds 25% utilization, it will move the coldest files to off-line storage
until the on-line usage is reduced to 20% utilization level. Only files that have
data on-line are eligible for migration. */
RULE 'MoveOffline' MIGRATE FROM POOL 'system'
THRESHOLD(25,20)
WEIGHT(weight_expression)
TO POOL 'mcstore'
WHERE(KB_ALLOCATED > 0) AND NOT(exclude_list)
/* Define default placement rule */
RULE 'Placement' SET POOL 'system'
RULE 'OpenRead'
EVENT 'OPEN_READ'
```

```
ACTION(System('/opt/ibm/MCStore/bin/mcstore recall -c -i ' || varchar(INODE) || ' -g '
|| varchar(GENERATION) || ' -s 0' || ' -f ' || varchar(FS_ID)) = 0)
WHERE(XATTR('dmapi.MCEA', 5, 1) == 'N')
RULE 'else' EVENT 'OPEN_READ' DIRECTORIES_PLUS
RULE 'OpenWrite'
EVENT 'OPEN_WRITE'
ACTION(System('/opt/ibm/MCStore/bin/mcstore recall -c -i ' || varchar(INODE) || ' -g '
|| varchar(GENERATION) || ' -s 0' || ' -f ' || varchar(FS_ID)) = 0)
WHERE(XATTR('dmapi.MCEA', 5, 1) == 'N')
RULE 'else' EVENT 'OPEN_WRITE' DIRECTORIES_PLUS
```

For more information on how to work with the external storage pools and related policies, see "Working with external storage pools" on page 330.

**Note:** Ensure that only a single instance of the policy is applied to migrate data to the external cloud storage pool. This avoids any potential locking issues that might arise due to multiple policy instances that try to work on the same set of files.

To ensure proper invocation of the policy on reaching threshold limits, see Threshold based migration using callbacks example.

In the sample policy, the 'OpenRead' & 'OpenWrite' rule sections represent the transparent recall of a migrated or non-resident file. Gateway software adds its own extended attributes (dmapi.MCEA) to each file it processes. Displacement 5 in the extended attributes indicate the resident state of the file. If it is 'N' (non-resident), the policy issues a recall request to bring back the data from the cloud storage to the local file system for the requested Read or Write operation.

| To apply a threshold policy to a file system, see "Using thresholds to migrate data between pools" on page 328.

IBM Spectrum Scale also enables administrators a way to define policies to identify the files for migration, and apply it immediately using the **mmapplypolicy** command. This is different from the threshold-based policies (which are applied by using the **mmchpolicy** command). Gateway service currently does not support parallelism in migrating files simultaneously, but parallelism in the **mmapplypolicy** command can be used to improve the overall Gateway throughput.

A sample command to apply a policy is given here:

```
mmapplypolicy gpfs0 -P <rules.file> -m 24 -B 100
```

where,

- *gpfs0* indicates the IBM Spectrum Scale system
- *-m* indicates the number of threads created and dispatched during policy execution phase.
- *-B* indicates the maximum number of files passed to each invocation of the EXEC script specified in the <rules.file>

**Note:** These two parameters (-m and -B) can be adjusted to improve the performance of massive migrations.

## Migrating files to the cloud storage tier

This topic provides a brief description on how to migrate files to the cloud storage tier by using Transparent Cloud Tiering.

**Note:** Before you try to migrate files to the cloud storage tier, ensure that a cloud storage account is created by using the **mmcloudgateway account create** command.

You can trigger migration of files from your file system to an external cloud storage tier either transparently or manually. Transparent migration is based on the policies that are applied on a file system. Data is automatically moved from the system pool to the configured external storage tier when the system pool reaches a certain threshold level. A file can be automatically migrated to/from cloud storage pool based on some characteristics of the file such as age, size, last access time, path. Alternatively, the user can manually migrate specific files or file sets to a cloud storage pool. For more information on policy-based migration, see "Applying a policy on a Transparent Cloud Tiering node" on page 597.

To manually trigger migration of the file *file1*, issue this command:

```
mmcloudgateway files migrate file1
```

The state of the file becomes **Non-resident** after it is successfully migrated to the cloud storage tier.

command. For more information on manually migrating files to the cloud storage tier, see **mmcloudgateway** command in the *IBM Spectrum Scale: Command and Programming Reference*.

## Recalling files from the cloud storage tier

This topic provides a brief description on how to recall files from the cloud storage tier by using Transparent Cloud Tiering.

You can trigger recall of files from the cloud storage tier either transparently or manually. Transparent recall is based on the policies that are applied on a file system. Data is automatically moved from the cloud storage tier to the system pool when the system pool reaches a certain threshold level. A file can be automatically recalled from the cloud storage tier based on some characteristics of the file such as age, size, last access time, path. Alternatively, the user can manually recall specific files or file sets. For more information on policy-based recall, see "Applying a policy on a Transparent Cloud Tiering node" on page 597.

**Note:** Like recalls in IBM Spectrum Archive and IBM Spectrum Protect for Space Management (HSM), Transparent Cloud Tiering recall would be filling the file with uncompressed data and the user would need to re-compress it by using **restripefs** or **restripefile** if so desired. Since we are positioning compression feature for cold data currently, the fact of a file being recalled means the file is no longer cold and leaving the file uncompressed would allow better performance for active files.

To manually trigger recall of the file *file1*, issue this command:unders

```
mmcloudgateway files recall file1
```

The state of the file becomes **Co-resident** after it is successfully recalled.

For more information on manually recalling files, see **mmcloudgateway** command in the *IBM Spectrum Scale: Command and Programming Reference*.

## Reconciling files between IBM Spectrum Scale file system and cloud storage tier

This topic describes how to reconcile files between IBM Spectrum Scale file systems and the cloud tier.

When you remove files from the file system, no policy is currently available to automatically remove the cloud objects that are created, which might cause orphan objects on the cloud.

To clean that up, it is recommended to periodically run the following command to keep the cloud in sync with the file system.

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

where,

- **Device** is the device name of the file system that is associated with the node class

If a migrated file is removed from the file system, reconcile removes the corresponding cloud objects and references that are contained in the cloud directory. Additionally, if multiple versions of a file are stored on the cloud, reconcile removes all older cloud versions (keeping the most recent). For example, if a file is migrated, then updated, and migrated again. In this case, two versions of the file are stored on the cloud. Reconcile removes the older version from the cloud. Reconcile also deletes cloud objects that are no longer referenced.

**Note:** Reconcile removes entries from the Cloud Directory that references deleted file system objects. The ability to restore accidentally deleted files is lost after reconciliation is run. Therefore, it is recommended that any files that need to be restored are restored before you run a reconcile. It is also recommended to run the reconciliation operation as a background activity during low load on the IBM Spectrum Scale cluster.

## Cleaning up files transferred to the cloud storage tier

This topic describes how to clean up files that are transferred to the cloud storage tier by using Transparent Cloud Tiering.

To do basic cleanup of objects that are transferred to the cloud object storage by using Transparent Cloud Tiering, issue a command according to this syntax:

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

where,

- **--recall-cloud-file**: When this option is specified, the files are recalled from the cloud storage before deleting them on the cloud. The status of the local files becomes resident after the operation.
- **--delete-local-file**: This option deletes both local files and the corresponding cloud object. There is no recall here.
- **--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.
- **--require-local-file**: This option removes the extended attribute 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.
- **--File**: This option can be used to process a file list similar to the one generated by the ILM policy.

The **mmcloudgateway files delete** command accepts files in GPFS file system as an input.

**Note:** You must run the **mmcloudgateway files reconcile** command after a delete operation. This is to ensure that the internal database is up-to-date after any delete operation. If the reconcile command is not done, the **mmcloudgateway files cloudList** command might show stale entries.

## Listing files migrated to the cloud storage tier

Even if the files are deleted from the file system after migration, you can generate a list of files that are migrated to the cloud storage tier. By using the file names, you can use the **mmcloudgateway restore** option to retrieve the files back from the cloud storage tier.

To list the files that are migrated to the cloud, issue a command according to this syntax:

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

For example, to list all files in the current directory, issue this command:

```
mmcloudgateway files cloudList --path /gpfs0/folder1
```

To list all files in all directories under the current directory, issue this command:

```
mmcloudgateway files cloudList --path /gpfs0/folder1 --recursive
```

To find all files named `myfile` in all directories under the current directory, issue this command:

```
mmcloudgateway files cloudList --path /gpfs0/folder1 --file myfile
```

To find all files named `myfile` in the current directory, issue this command:

```
mmcloudgateway files cloudList --path /gpfs0/folder1 --depth 0 --file myfile
```

To display information about all versions of file `myfile` in current directory, issue this command:

```
mmcloudgateway files cloudList --file-versions myfile
```

## Restoring files

This topic provides a brief description on how to restore files from the cloud storage tier if the original files are deleted from the GFPS file system.

This option provides a non-optimized (emergency) support for manually restoring files from the cloud storage tier if the original stub files on the GPFS file system are deleted.

To restore files, issue a command according to this syntax:

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

**Note:** To restore files, you need to provide the file names. You can retrieve the names of the deleted files by using the **mmcloudgateway files cloudList** command. If you perform any reconcile operation after deletion of the files from the local file system, the records of the deleted files are removed from the database and hence the files cannot be restored. Therefore, ensure that you do not perform any reconcile operation between deletion of the files and the restore operation.
For information on the description of the parameters, see the **mmcloudgateway** command in *IBM Spectrum Scale: Command and Programming Reference*.

---

## Deleting a file system association

This topic describes the procedure for deleting the file system association for Transparent Cloud Tiering.

To break the connection between Transparent Cloud Tiering and the file system, issue a command according to this syntax:

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

For example, to delete the association between the file system /gpfs0 and the node class CloudNodeClass, issue this command:

```
mmcloudgateway filesystem delete --cloud-nodeclass TCTNodeClass --filesystem /gpfs
```

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.
```

**Note:** This command does not delete any cloud objects, but simply breaks the connection. Data migrated to the cloud object storage remains intact.

## Deleting a cloud storage account

You can delete a cloud storage account when you want to move from one cloud account to another. For example, you have used Amazon S3 for testing purposes and then want to move to IBM Cloud Object Storage for production. In this case, you can delete the S3 account and then create a new IBM Cloud Object Storage account.

**Note:** Before deleting a cloud account, ensure that you recall all the data that is migrated to the cloud.

To delete a cloud storage account, issue a command according to this syntax:

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

For example, to delete the cloud account, *cloudtest*, for the node class, *TCTNodeClass*, issue this command:

```
mmcloudgateway account delete --cloud-nodeclass TCTNodeClass --cloud-name cloudtest
```

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.
```

## Manual recovery of Transparent Cloud Tiering database

Transparent Cloud Tiering uses a database called "cloud directory" to store a list and versions of files that are migrated to the cloud. Any issues in this database might lead to undesired results. If the database is corrupted or has been accidentally deleted, you can reconstruct it from automatic backups of this database that are kept on the cloud.

You need to perform a recovery of the database when Transparent Cloud Tiering produces any of the following messages in the logs:

- The cloud directory database for file system `/dev/gpfs0` could not be found. Manual recovery is necessary.
- The directory service for file system `/dev/gpfs0` is not ready for use. Manual recovery is necessary.
- The cloud directory database for file system `/dev/gpfs0` is corrupted. Manual recovery is necessary.

To perform a manual recovery, issue a command according to this syntax:

```
mmcloudgateway files rebuildDB filesystem
```

where,

`filesystem` is the device name of the file system whose database is corrupted and which is in need of manual recovery.

For example, if you want to recover the database associated with the file system, `/dev/gpfs0`, issue this command:

```
mmcloudgateway files rebuildDB /dev/gpfs0
```

## Known limitations of Transparent Cloud Tiering

This topic describes the limitations that are identified for Transparent Cloud Tiering.

**mmcloudgateway files migrate * on a parent folder does not move all files within the subfolders**
Running the **mmcloudgateway files migrate** command to migrate all files (including the files within the subfolders) does not migrate all files within subfolders. It only migrates leaf files within the current folder, from which the migrate command is issued. The migrate process skips the subfolders, by displaying the following warning message:

```
MCSTG00051E: File is not a regular file. Migration requests only support regular files.
error processing /<file-system-mount>/<folder1>/<folder-2>....
```

To migrate all files (including files within the subfolders) in one go, issue this command:

```
find <gpfs-mountpoint-folder-or-subfolder> -type f –exec mmcloudgateway files migrate {} +
```

This command passes the entire list of files to a single migrate process in the background as follows:

```
mmcloudgateway files migrate <file1> <file2> <sub-folder1/file1> <sub-folder2/file1> ......
```

**Migrating Transparent Cloud Tiering specific configuration to cloud storage might lead to issues**
While you move data to an external cloud storage tier, it is recommended not to migrate files within the Transparent Cloud Tiering internal folder (.mcstore folder within the configured GPFS file system) to cloud storage. It might lead to undesirable behavior for the Transparent Cloud Tiering service. To address this issue, include the EXCLUDE directive in the migration policy.

Refer to the `/opt/ibm/MCStore/samples` folder to view sample policies that can be customized as per your environment and applied on the file system managed by Transparent Cloud Tiering.

**Running mmcloudgateway files delete on multiple files**
Trying to remove multiple files in one go with the **mmcloudgateway files delete –d *** fails with a **NullPointerException**. This happens while you clean up the cloud metrics. Issue this command to remove the cloud objects:

```
find <gpfs-file-system> -type f -exec mmcloudgateway files delete -d {} \;
```

**Range reads from the cloud object storage is not supported for transparent recall.**
When a file is transparently recalled, the file is entirely recalled.

**Policy-based migrations**
Policy-based migrations should be started only from Transparent Cloud Tiering server nodes. Client nodes should be used only for manual migration.

**https support for pre-test**
The **mmcloudgateway account pre-test** command does not provide support for https endpoints. For https support, use the **mmcloudgateway account create** command.

For current limitations and restrictions, see IBM Spectrum Scale FAQs.

For more information, see the topic *Interoperability of Transparent Cloud Tiering with other IBM Spectrum Scale features* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

# Chapter 37. Highly-available write cache (HAWC)

Highly-available write cache (HAWC) reduces the latency of small write requests by initially hardening data in a non-volatile fast storage device prior to writing it back to the backend storage system.

## Overview and benefits

Current disk drive systems are optimized for large streaming writes, but many workloads such as VMs and databases consist of many small write requests, which do not perform well with disk drive systems. To improve the performance of small writes, storage controllers buffer write requests in non-volatile memory before writing them to storage. This works well for some workloads, but the amount of NVRAM is typically quite small and can therefore not scale to large workloads.

The goal of HAWC is to improve the efficiency of small write requests by absorbing them in any nonvolatile fast storage device such as SSDs, Flash-backed DIMMs, or Flash DIMMs. Once the dirty data is hardened, GPFS can immediately respond to the application write request, greatly reducing write latency. GPFS can then flush the dirty data to the backend storage in the background.

By first buffering write requests, HAWC allows small writes to be gathered into larger chunks in the page pool before they are written back to storage. This has the potential to improve performance as well by increasing the average amount of data that GPFS writes back to disk at a time.

Further, when GPFS writes a data range smaller than a full block size to a block for the first time, the block must first be fully initialized. Without HAWC, GPFS does this by writing zeroes to the block at the time of the first write request. This increases the write latency since a small write request was converted into a large write request (for example, a 4K write request turns into a 1MB write request). With HAWC, this initialization can be delayed until after GPFS responds to the write request, or simply avoided altogether if the application subsequently writes the entire block.

To buffer the dirty data, HAWC hardens write data in the GPFS recovery log. This means that with HAWC, the recovery log must be stored on a fast storage device because if the storage device on which the recovery log resides is the same as the data device, HAWC will decrease performance by writing data twice to the same device. By hardening data in the recovery log, all incoming requests are transformed into sequential operations to the log. In addition, it is important to note that applications never read data from the recovery log, since all data that is hardened in the recovery log is always kept in the page pool. The dirty data in the log is only accessed during file system recovery due to improper shutdown of one or more mounted instances of a GPFS file system.

The maximum size of an individual write that can be placed in HAWC is currently limited to 64KB. This limit has been set for several reasons, including the following:

- The benefit of writing data to fast storage decreases as the request size increases.
- Fast storage is typically limited to a much smaller capacity than disk subsystems.
- Each GPFS recovery log is currently limited to 1GB. Every file system and client pair has a unique recovery log. This means that for each file system, the size of HAWC scales linearly with every additional GPFS client. For example, with 2 file systems and 10 clients, there would be 20 recovery logs used by HAWC to harden data.

Note that combining the use of HAWC with LROC allows GPFS to leverage fast storage on application reads and writes.

## Applications that can benefit from HAWC

Typically, it is recommended to place GPFS metadata in a storage pool consisting of fast storage devices such as SSDs. Storing GPFS recovery logs in fast storage improves the performance of metadata-intensive workloads where the recovery log is heavily used and when GPFS is configured to replicate data.

With HAWC, storing the recovery log in fast storage has the added benefit that workloads that experience bursts of small and synchronous write requests (no matter if they are random or sequential) will also be hardened in the fast storage. Well-known applications that exhibit this type of write behavior include VMs, databases, and log generation.

Since the characteristics of fast storage vary greatly, users should evaluate their application workload with HAWC in their storage configuration to ensure a benefit is achieved. In general, however, speedups should be seen in any environment that either currently lacks fast storage or has very limited (and non-scalable) amounts of fast storage.

## Restrictions and tuning recommendations for HAWC

When enabling HAWC, take the following restrictions and tuning recommendations into consideration:

**Ping pong recovery log buffers**

Ping pong recovery log buffers should *not* be enabled when the recovery log is stored on storage devices that can gracefully write data upon power failure. This includes SSDs, NVRAM, storage controllers, RAID controllers among others.

Ping pong buffers are only needed to avoid data corruption when the recovery log is stored directly on disk. They place log data in two separate locations on disk to avoid loss of that data if a sector becomes unavailable. Writing to two separate locations creates additional overhead that is exacerbated by HAWC due to the large amount of data it places in the recovery log.

In general, it is not recommended to use HAWC with any storage device that would require ping pong buffers to be enabled because it doubles the amount of data that must be written before GPFS can respond to a application write request.

To disable log buffers, run the following command:

```
mmchconfig logPingPongSector=no
```

**Recovery log size**

The size of the recovery log defaults to a very small value (less than 16MB), which is not sufficient space to buffer HAWC data. Therefore, it is recommended to increase the size of the log at least to 128M or larger (1GB maximum). However, the effect of a larger recovery log is that upon node failure, more data must be recovered into the storage system, which will increase the time it takes to recover. It is important to take this into account because applications will not be able to access data in the file system while recovery is running.

**Encryption**

Encrypted data is never stored in the recovery log, but instead follows the pre-GPFS 4.1.0.4 semantics for synchronous writes even if the HAWC threshold is set to a value greater than 0.

**Small files and directory blocks**

HAWC does not change the following:

- write behavior of small files when the data is placed in the inode itself
- write behavior of directory blocks or other metadata

# Using HAWC

To use HAWC effectively, you must enable it with an appropriate threshold, ensure the correct storage of the GPFS recovery log, and properly perform other administrative tasks.

## Enabling HAWC

To enable HAWC, set the write cache threshold for the file system to any value between 4K and 64K (specified in multiples of 4KB), as shown in the following examples:

```
mmchfs gpfsA --write-cache-threshold 32K
```

or

```
mmcrfs /gpfs/gpfsB /dev/gpfsB -F ./deiskdef2.txt -B1M --write-cache-threshold 32K
```

Once HAWC has been enabled, all synchronous write requests smaller or equal to the threshold will be placed in the recovery log, and a response will be immediately returned to the application. All synchronous write requests greater than the threshold will be written directly to the primary storage system as usual.

## Methods to ensure the use of the recovery log on fast storage

There are two methods for storing the GPFS recovery log. The method to choose depends on the storage architecture in which HAWC will be used.

Currently, Method 1 is the easier and more common method because it uses a centralized fast storage device, already common in many existing GPFS storage architectures. Method 2 enables the recovery log to be stored within the GPFS clients themselves, which can reduce latency, but it requires fast storage within every GPFS client node.

In either case, proper setup is important to ensure that written data will survive node or disk failures and will improve the performance of small synchronous writes.

**Method 1. Storing GPFS metadata on highly available storage appliances**

In this method, GPFS data is stored on a centralized fast storage device such as any of the following:

- Storage controller with SSDs
- FlashSystems
- GPFS Storage Server (GSS) with SSDs

This method is relatively simple to set up, as NSDs can be specified to contain only metadata upon their creation. As always, the metadata storage pool can be replicated for higher availability.

**Method 2 . Storing the GPFS recovery log on GPFS client nodes**

This method calls for the following setup:

- GPFS clients (or a subset of them) have fast storage devices installed in them.
- The GPFS recovery log is stored in the NVRAM on the GPFS clients (separate from GPFS metadata).

  To do this, specify the `system.log` pool for the NVRAM NSDs on the GPFS clients at file system creation time or when adding disks to the file system. When those NSDs are used to create the file system, the recovery log will be placed in the `system.log` pool (which is now defined to be on the GPFS clients) instead of in the system pool.
- Because GPFS clients are not highly available (meaning that if they failed, the data stored on them would become unavailable), the `system.log` pool must be replicated.

To specify a replication factor for the `system.log` pool, the **--log-replicas** option must be used. The **--log-replicas** option is intended to be used in conjunction with the `system.log` pool feature to place log files in a separate pool with replication different from other metadata in the system pool.

**Note:** Log replication can be changed dynamically with the **mmchfs** command followed by the **mmrestripefs** command. There is no separate option for maximum log replication; therefore, on a file system created with a single replica of the recovery log (**--log-replicas=1**), it is only possible to enable log replication if the file system was created with **-M 2** or **-M 3**.

## Additional administrative tasks

Using HAWC may also require the following administrative tasks:

**Adding or removing disks from the `system.log` pool**
The **mmrestripefs -b** command will restripe log files. Therefore, if you have enough NVRAM for all the logs, but too much of the log data ended up on a small number of disks because of the way the file system was created, **mmrestripefs** can be used to rebalance the data across the nodes.

**Handling node/disk failures in the `system.log` pool**
If the log is replicated, this command will ensure that data is re-replicated automatically when a node/disk fails:

```
mmchconfig restripeOnDiskFailure=yes -i
```

You can use this command to fine-tune how quickly the data will be replicated after failure (to avoid re-replication in case of node reboots, for example):

```
mmchconfig metadataDiskWaitTimeForRecovery=seconds
```

The default value for **metadataDiskWaitTimeForRecovery** is 300 seconds.

**Using HAWC with Existing File Systems**
To enable HAWC with an existing file system (once the entire cluster has been brought up to the latest file system and configuration version using **mmchconfig release=LATEST**), follow one or more of the following guidelines.

- If the metadata pool is already stored on a fast storage device, simply increase the size of the recovery log to at least 128M prior to setting the HAWC threshold by running the following command:

  ```
  mmchfs Device -L LogFileSize
  ```

- If the metadata pool is not already on a fast storage device, it must first be migrated to a fast storage device. For more information on storage pools, see “Managing storage pools” on page 296. Once the file system metadata is stored on fast storage devices, increase the size of the recovery log (as explained previously) and set the HAWC threshold.

# Chapter 38. Local read-only cache

Many applications benefit greatly from large local caches. Not only is the data available with very low latency, but the cache hit serves to reduce the load on the shared network and on the backend storage itself, thus benefiting all nodes, even those without large caches.

Local solid state disks (SSDs) provide an economical way to create very large caches. The SSD cache serves as an extension to the local buffer pool. As user data or metadata is evicted from the buffer pool in memory, it can be stored in the local cache. A subsequent access will retrieve the data from the local cache, rather than from the home location. The data stored in the local cache, like data stored in memory, remains consistent. If a conflicting access occurs, the data is invalidated from all caches. In a like manner, if a node is restarted, all data stored in the cache is discarded.

In theory, any data or metadata may be stored in the local SSD cache, but the cache works best for small random reads where latency is a primary concern. Since the local cache typically offers less bandwidth than the backend storage, it may be unsuitable for large sequential reads. The configuration options provide controls over what is stored in the cache. The default settings are targeted at small random I/O.

The local read-only cache function is disabled by default. To enable it, the administrator must define an NSD to be used by local read-only cache. The local read-only cache disk is expected to be a solid state disk (SSD) accessible via SCSI. The device as defined as a standard NSD by **mmcrnsd**, but the **DiskUsage** is set to **localCache**. The NSD must have a primary server and is not allowed to have other servers. The primary server must be the node where the physical local read-only cache device is installed. The device is *not* exported to other nodes in the cluster. The storage pool and failure group defined for NSD are ignored and should be set to null. The **mmcrnsd** command creates a unique NSD ID and name for the device and updates sector 2 to indicate that it is an NSD.

The minimum size of a local read-only cache device is 4 GB. The local read-only cache requires memory equal to 1% of the capacity of the local read-only cache device.

Once the local read-only cache disk is defined, the daemon code at the primary server node is automatically told to do device discovery. The daemon will detect that **localCache** is defined for its use and will determine the mapping to the local device. The daemon then informs the local read-only cache code to begin using the device for caching. Currently, there is a limit of four **localCache** devices per node. Note that the daemon code does not need to be restarted to begin using the cache.

The local read-only cache device may be deleted by using the **mmdelnsd** command. Both **mmcrnsd** and **mmdelnsd** may be issued while the daemon is running with file systems mounted and online. The call to delete the NSD first informs the daemon that the device is being deleted, which removes it from the list of active local read-only cache devices. Any data cached on the device is immediately lost, but data cached on other local read-only cache devices is unaffected. Once the **mmdelnsd** command completes, the underlying SSD may be physically removed from the node.

The NSD name for the local read-only cache device may not be used in any other GPFS commands, such as **mmcrfs**, **mmadddisk**, **mmrpldisk**, **mmchdisk** or **mmchnsd**. The device will be shown by **mmlsnsd** as a **localCache**.

# Chapter 39. Miscellaneous advanced administration topics

The following topics provide information about miscellaneous advanced administration tasks:



## Changing IP addresses and host names

GPFS assumes that IP addresses and host names remain constant. In the rare event that such a change becomes necessary or is inadvertently introduced by reinstalling a node with a disk image from a different node for example, follow the steps in this topic.

If all of the nodes in the cluster are affected and all the conditions in Step 2 following are met:

1. Run the **mmshutdown -a** command to stop GPFS on all nodes.
2. Using the documented procedures for the operating system, add the new host names or IP addresses but do not remove the old ones yet. This can be achieved, for example, by creating temporary alias entries in **/etc/hosts**. Do not restart the nodes until the **mmchnode** command in Step 3 is executed successfully. If any of these conditions cannot be met, use the alternate procedure described in this section.
3. Update the nodes that the cluster uses as the administration interface node and the daemon interface node, if necessary. To update these values, run the **mmchnode** command with the **--admin-interface** and the **--daemon-interface** options.

   **Note:** 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.
4. If the IP addresses over which the subnet attribute is defined are changed, you must update your configuration by running the **mmchconfig** command with the **subnets** attribute.
5. Start GPFS on all nodes with **mmstartup -a**.
6. Remove the unneeded old host names and IP addresses.

If only a subset of the nodes are affected, it may be easier to make the changes using these steps:

1. Before any of the host names or IP addresses are changed:
   - Use the **mmshutdown** command to stop GPFS on all affected nodes.
   - If the host names or IP addresses of the primary or secondary GPFS cluster configuration server nodes must change, use the **mmchcluster** command to specify another node to serve as the primary or secondary GPFS cluster configuration server.
   - If the host names or IP addresses of an NSD server node must change, temporarily remove the node from being a server with the **mmchnsd** command. Then, after the node has been added back to the cluster, use the **mmchnsd** command to change the NSDs to their original configuration. Use the **mmlsnsd** command to obtain the NSD server node names.
   - Use the **mmdelnode** command to delete all affected nodes from the GPFS cluster.
2. Change the node names and IP addresses using the documented procedures for the operating system.

3. If the IP addresses over which the subnet attribute is defined are changed, you need to update your configuration by using the **mmchconfig** command with the **subnets** attribute.
4. Issue the **mmaddnode** command to restore the nodes to the GPFS cluster.
5. If necessary, use the **mmchcluster** and **mmchnsd** commands to restore the original configuration and the NSD servers.

## Enabling a cluster for IPv6

For newly created clusters, if any of the specified node interfaces on the **mmcrcluster** command resolves to an IPv6 address, the cluster is automatically enabled for IPv6. For existing IPv4-based clusters, follow one of the procedures that described in this section.

If you are performing the procedure during a scheduled maintenance window and GPFS can be shut down on all of the nodes in the cluster, run the command:

```
mmchconfig enableIPv6=yes
```

After the command finishes successfully, you can start adding new nodes with IPv6 addresses.

If it is not possible to shut down GPFS on all of the nodes at the same time, run the command:

```
mmchconfig enableIPv6=prepare
```

The next step is to restart GPFS on each of the nodes so that they can pick up the new configuration setting. This can be done one node at a time when it is convenient. To verify that a particular node has been refreshed, run:

```
mmdiag --config | grep enableIPv6
```

The reported value should be 1.

Once all of the nodes have been recycled in this manner, run the command:

```
mmchconfig enableIPv6=commit
```

This command will only succeed when all GPFS daemons have been refreshed. Once this operation succeeds, you can start adding new nodes with IPv6 addresses.

To convert an existing node from an IPv4 to an IPv6 interface, use one of the procedures described in "Changing IP addresses and host names" on page 611.

## Using multiple token servers

Distributed locking, allowing GPFS to maintain a consistent view of the file system, is implemented using token-based lock management. Associated with every lockable object is a token.

Before a lock on an object can be granted to a thread on a particular node, the lock manager on that node must obtain a token from the token server. The total number of token manager nodes depends on the number of manager nodes defined in the cluster.

When a file system is first mounted, the file system manager is the only token server for the file system. Once the number of external mounts exceeds one, the file system manager appoints all the other manager nodes defined in the cluster to share the token server load. Once the token state has been distributed, it remains distributed until all external mounts have gone away. The only nodes that are eligible to become token manager nodes are those designated as manager nodes.

The number of files for which tokens can be retained on a manager node is restricted by the values of the **maxFilesToCache** and **maxStatCache** configuration parameters. Distributing the tokens across multiple

token manager nodes allows more tokens to be managed or retained concurrently, improving performance in situations where many lockable objects are accessed concurrently.

## Exporting file system definitions between clusters

You can export a GPFS file system definition from one GPFS cluster to another.

To export file system definitions between clusters, follow these steps:

1. Ensure that all disks in all GPFS file systems to be migrated are in working order by issuing the **mmlsdisk** command. Verify that the disk status is ready and availability is up. If not, correct any problems and reissue the **mmlsdisk** command before continuing.
2. Stop all user activity in the file systems.
3. Follow any local administrative backup procedures to provide for protection of your file system data in the event of a failure.
4. Cleanly unmount all affected GPFS file systems. Do not use force unmount.
5. Export the GPFS file system definitions by issuing the **mmexportfs** command. This command creates the configuration output file *ExportDataFile* with all relevant file system and disk information. Retain this file as it is required when issuing the **mmimportfs** command to import your file systems into the new cluster. Depending on whether you are exporting a single file system or all of the file systems in the cluster, issue:

   ```
   mmexportfs fileSystemName -o ExportDataFile
   ```

   or

   ```
   mmexportfs all -o ExportDataFile
   ```
6. Ensure that the file system disks from the old GPFS cluster are properly connected, and are online and available to be accessed from appropriate nodes of the new GPFS cluster.
7. To complete the movement of your file systems to the new cluster using the configuration file created in Step 5, issue one of these commands, depending on whether you are importing a single file system or all of the file systems in the cluster:

   ```
   mmimportfs fileSystemName -i ExportDataFile
   ```

   or

   ```
   mmimportfs all -i ExportDataFile
   ```

## GPFS port usage

The nodes in a GPFS cluster communicate with each other using the TCP/IP protocol. The port number used by the main GPFS daemon (**mmfsd**) is controlled with the **tscTcpPort** configuration parameter. The default port number is 1191.

You can specify a different port number using the **mmchconfig** command:

```
mmchconfig tscTcpPort=PortNumber
```

When the main GPFS daemon (**mmfsd**) is not running on the primary and backup configuration server nodes, a separate service (**mmsdrserv**) is used to provide access to the configuration data to the rest of the nodes in the cluster. The port number used for this purpose is controlled with the **mmsdrservPort** parameter. By default, **mmsdrserv** uses the same port number as the one assigned to the main GPFS daemon. If you change the daemon port number, you must specify the same port number for **mmsdrserv** using the following command:

```
mmchconfig mmsdrservPort=PortNumber
```

Do not change the **mmsdrserv** port number to a number different from that of the daemon port number.

Certain commands (**mmadddisk**, **mmchmgr**, and so on) require an additional socket to be created for the duration of the command. The port numbers assigned to these temporary sockets are controlled with the **tscCmdPortRange** configuration parameter. If an explicit range is not specified, the port number is dynamically assigned by the operating system from the range of ephemeral port numbers. If you want to restrict the range of ports used by GPFS commands, use the **mmchconfig** command:

```
mmchconfig tscCmdPortRange=LowNumber-HighNumber
```

In a remote cluster setup, if GPFS on the remote cluster is configured to use a port number other than the default, you have to specify the port number to be used with the **mmremotecluster** command:

```
mmremotecluster update ClusterName -n tcpPort=PortNumber,Node,Node...
```

Table 58 provides GPFS port usage information:

*Table 58. GPFS port usage*

| Descriptor | Explanation |
|---|---|
| Service provider | GPFS |
| Service name | **mmfsd**<br>**mmsdrserv** |
| Port number | 1191<br><br>While executing certain commands, GPFS may need to create additional sockets whose dynamic port numbers are assigned by the operating system. Such sockets are used by commands to exchange data with GPFS daemons running on other nodes. The port numbers that are used correspond to the ephemeral ports of the operating system.<br><br>To control which ports are used by the commands (so that firewall rules can be written to allow incoming traffic only on those ports), you can restrict the port range to a specific range by setting the **tscCmdPortRange** configuration variable. |
| Protocols | TCP/IP |
| Source port range | The source port range is chosen by the operating system on the client side. |
| Is the service name/number pair in the default /etc/services file shipped with AIX and Linux distributions? | See the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html). |
| Is the service name/number pair added to /etc/services by a product? | No |
| Binaries that listen on the ports | `/usr/lpp/mmfs/bin/mmfsd`<br>`/usr/lpp/mmfs/bin/mmsdrserv` |

*Table 58. GPFS port usage (continued)*

| Descriptor | Explanation |
|---|---|
| Can the service be configured to use a different port? | Yes. To change the main port used by GPFS, use:<br>`mmchconfig tscTcpPort=PortNumber`<br><br>**Note:** If you change the main port (daemon port) number, you must change the **mmsdrserv** port to the same number.<br><br>To change the **mmsdrserv** port number to match the daemon port number, use:<br>`mmchconfig mmsdrservPort=PortNumber`<br><br>To change the range of port numbers used for command execution, use:<br>`mmchconfig tscCmdPortRange=LowNumber-HighNumber`<br><br>To specify a port number when connecting to remote clusters, use the **mmremotecluster** command. |
| When is the service required? What depends on the service? | On the GPFS primary and secondary cluster configuration servers, either **mmsdrserv** or **mmfsd** needs to be running at all times to provide access to GPFS configuration data to the rest of the cluster. On other nodes, **mmfsd** must be running in order to mount a GPFS file system. Depending on the GPFS configuration, a node either has to be a member of the GPFS cluster or possess an authorized SSL key in order to establish a connection. |
| When the daemon starts and its port is already in use (for example, another resource has bound to it already), how does the daemon behave? | The daemon shuts down and tries to start over again.<br><br>Most GPFS daemon down error messages are in the **mmfs.log.previous** log for the instance that failed. If the daemon restarted, it generates a new **mmfs.log.latest** log.<br><br>Begin problem determination for these errors by examining the operating system error log. GPFS records file system or disk failures using the error logging facility provided by the operating system: **syslog** facility on Linux and **errpt** facility on AIX.<br><br>See the *IBM Spectrum Scale: Problem Determination Guide* for further information. |
| Is there an administrator interface to query the daemon and have it report its port number? | Yes; run this command:<br>**`mmlsconfig tscTcpPort`** |
| Is the service/port registered with the Internet Assigned Numbers Authority (IANA)? | Yes<br>`gpfs 1191/tcp General Parallel File System`<br>`gpfs 1191/udp General Parallel File System`<br>`# Dave Craft <gpfs@ibm.com>`<br>`November 2004` |

**Note:** Ports configured for **gpfsClusterRemoteShellCommand** (for example, **ssh**) and ICMP (ping) also must be unblocked in the firewall for GPFS to function properly

# Securing the IBM Spectrum Scale system using firewall

The IBM Spectrum Scale system is an open system where the customer can interact with the system through other third-party interfaces like MMC, web applications, and so on. The customer also has root access to the system just like any Linux server administrator. Firewalls that are associated with open systems are specific to deployments, operating systems, and it vary from customer to customer. It is the responsibility of the system administrator or Lab Service (LBS) to set the firewall accordingly; similar to what Linux distributions do today. This section provides recommendations to set up firewall to secure the IBM Spectrum Scale protocol nodes.

For firewall recommendations for Transparent Cloud Tiering, see the *Firewall recommendations for Transparent Cloud Tiering* topic in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

# Firewall recommendations for the IBM Spectrum Scale installation

It is recommended to allow connection only from the IBM Spectrum Scale cluster node IPs (internal IPs and protocol IPs) on port 8889 and block all other external connections on this port during the installation process.

Installer uses the following Chef port during IBM Spectrum Scale installation.

*Table 59. Recommended port numbers that can be used for installation*

| Port Number | Protocol | Service Name | Components involved in communication |
|---|---|---|---|
| 8889 | TCP | Chef | Intra-cluster and installer server |
| 10080 | TCP | Repository | Intra-cluster and installer server |

The port that is used during the installation (8889) can be blocked when the installation is over. You can get the list of protocol IPs by using the **mmlscluster --ces** command. Use the **mmlscluster** command to get the list of all internal IPs.

# Firewall recommendations for internal communication among nodes

The IBM Spectrum Scale system uses the following ports for internal communication among various IBM Spectrum Scale nodes.

*Table 60. Recommended port numbers that can be used for internal communication*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 1191 | TCP | GPFS | Intra-cluster |
| 22 | TCP | SSH | Clients who are accessing the system |

The following are the recommendations for securing internal communications among IBM Spectrum Scale nodes:

- Allow connection only to the GPFS cluster node IPs (internal IPs and protocol node IPs) on port 1191. Block all other external connections on this port. Use the **mmlscluster --ces** command to get the list of protocol node IP and use the **mmlscluster** command to get the list of IPs of internal nodes.
- Allow all external communications request that are coming from the admin or management network and IBM Spectrum Scale internal IPs on port 22.

- Certain commands such as **mmadddisk**, **mmchmgr**, and so on require an extra socket to be created for the duration of the command. The port numbers that are assigned to these temporary sockets are controlled with the **tscCmdPortRange** configuration parameter. If an explicit range is not specified, the port number is dynamically assigned by the operating system from the range of ephemeral port numbers. It is highly recommended to set the port range. For more information on how to set the port range, see "GPFS port usage" on page 613.

## Firewall recommendations for protocol access

It is recommended to use certain port numbers to secure the protocol data transfer.

### Recommendations for NFS access

The following table provides the list of static ports that are used for NFS data I/O.

*Table 61. Recommended port numbers for NFS access*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 2049 | TCP and UDP | NFSV4 or NFSV3 | NFS clients and IBM Spectrum Scale protocol node |
| 111 | TCP and UDP | RPC (required only by NFSV3) | NFS clients and IBM Spectrum Scale protocol node |
| User-defined static port | TCP and UDP | STATD (required only by NFSV3) | NFS clients and IBM Spectrum Scale protocol node |
| User-defined static port | TCP and UDP | MNT (required only by NFSV3) | NFS clients and IBM Spectrum Scale protocol node |
| User-defined static port | TCP and UDP | NLM (required only by NFSV3) | NFS clients and IBM Spectrum Scale protocol node |
| User-defined static port | TCP and UDP | RQUOTA (required by NFSV3)<br><br>RQUOTA (required by both NFSV3 and NFSV4) | NFS clients and IBM Spectrum Scale protocol node |

**Note:** The NFSV3 uses the dynamic ports for NLM, MNT, and STATD services. When NFSV4 server is used with the firewall, these services must be configured with static ports.

The following recommendations are applicable:

- Set static ports for MNT, NLM, and STATD services that are required by the NFSV3 server by using **mmnfs configuration change** command. Allow TCP and UDP port 2049 to use the protocol node IPs. For example:

  ```
  mmnfs configuration change MNT_PORT=32767:NLM_PORT=32769:RQUOTA_PORT=32768:STATD_PORT=32765
  ```

- Allow all external communications on TCP and UDP port 111 by using the protocol node IPs.
- Allow all external communications on TCP and UDP port that is specified with **mmnfs configuration change** for MNT and NLM ports.

## Recommendations for SMB access

Samba uses the following ports for the secure access.

*Table 62. Recommended port numbers for SMB access*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 445 | TCP | Samba | SMB clients and IBM Spectrum Scale protocol node |
| 4379 | TCP | CTDB | Inter-protocol node |

The following recommendations are applicable for the SMB access:

- Allow the access request that is coming from the data network and admin and management network on port 445 using the protocol node IPs. You can get the list of protocol node IPs by using the **mmlscluster --ces** command.
- Allow connection only to the requests that are coming from the IBM Spectrum Scale cluster node IPs (internal IPs and protocol node IPs) on port 4379. Block all other external connections on this port. Use the **mmlscluster** command to get the list of cluster node IPs.

## Object port configuration

**Note:** IBM Spectrum Scale is configured with the ports listed here. Changing ports requires updating configuration files, Keystone endpoint definitions, and SELinux rules. This must be done only after careful planning.

The following table lists the ports configured for object access.

*Table 63. Port numbers for object access*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 8080 | TCP | Object Storage Proxy | Object clients and IBM Spectrum Scale protocol node |
| 6200 | TCP | Object Storage (local account server) | Local host |
| 6201 | TCP | Object Storage (local container server) | Local host |
| 6202 | TCP | Object Storage (local object server) | Local host |
| 6203 | TCP | Object Storage (object server for unified file and object access) | Local host |
| 11211 | TCP and UDP | Memcached (local) | Local host |

The following ports are configured for securing object access:

- Allow all external communications on TCP port 8080 (Object Storage proxy).
- Allow connection only from the IBM Spectrum Scale cluster node IPs (internal IPs and protocol node IPs) on ports 6200, 6201, 6202, 6203, and 11211. Block all other external connections on this port.

Shell access by non-root users must be restricted on IBM Spectrum Scale protocol nodes where the object services are running to prevent unauthorized access to object data.

**Note:** The reason for these restrictions is that because there is no authentication of requests made on ports 6200, 6201, 6202, and 6203, it is critical to ensure that these ports are protected from access by unauthorized clients.

**Port usage for object authentication**

You can configure either an external or internal Keystone server to manage the authentication requests. Keystone uses the following ports:

*Table 64. Port numbers for object authentication*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 5000 | TCP | Keystone Public | Authentication clients and object clients |
| 35357 | TCP | Keystone Internal/Admin | Authentication and object clients and Keystone administrator |

These ports are applicable only if keystone is hosted internally on the IBM Spectrum Scale system. The following port usage is applicable:

- Allow all external communication requests that are coming from the admin or management network and IBM Spectrum Scale internal IPs on port 35357.
- Allow all external communication requests that are coming from clients to IBM Spectrum Scale for object storage on port 5000. Block all other external connections on this port.

**Port usage to connect to the Postgres database for object protocol**

The Postgres database server for object protocol is configured to use the following port:

*Table 65. Port numbers for Postgres database for object protocol*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 5431 | TCP and UDP | postgresql-obj | Inter-protocol nodes |

It is recommended to allow connection only from Cluster node IPs (Internal ips and Protocol node IPs) on port 5431. Block all other communication requests on this port.

**Note:** The Postgres instance used by the object protocol uses port 5431. This is different from the default port to avoid conflict with other Postgres instances that might be on the system including the instance for IBM Spectrum Scale GUI.

## Consolidated list of recommended ports that are used for installation, internal communication, and protocol access

The following table provides a consolidated list of recommended ports and firewall rules.

*Table 66. Consolidated list of recommended ports for different functions*

| Function | Dependent network service names | External ports that are used for file and object access | Internal ports that are used for inter-cluster communication | UDP / TCP | Nodes for which the rules are applicable |
|---|---|---|---|---|---|
| Installer | Chef | N/A | 8889 (chef)<br>10080 (repo) | TCP | GPFS server, NSD server, protocol nodes |
| GPFS (internal communication) | GPFS | N/A | 1191 (GPFS)<br>60000-61000 for tscCmdPortRange<br>22 for SSH | TCP and UDP<br>TCP only for 22 | GPFS server, NSD server, protocol nodes |
| SMB | gpfs-smb.service<br>gpfs-ctdb.service<br>rpc.statd | 445 | 4379 (CTDB) | TCP | Protocol nodes only |

*Table 66. Consolidated list of recommended ports for different functions (continued)*

| Function | Dependent network service names | External ports that are used for file and object access | Internal ports that are used for inter-cluster communication | UDP / TCP | Nodes for which the rules are applicable |
|---|---|---|---|---|---|
| NFS | ganesha.nfsd<br><br>rpcbind<br><br>rpc.statd | 2049 (NFS_PORT - required by both NFSV3 and NFSV4)<br><br>2049 (NFS_PORT - required only by NFSV3)<br><br>111 (RPC - required only by NFSV3)<br><br>32765 (STATD_PORT)<br><br>32767 (MNT_PORT - required only by NFSV3)<br><br>32768 (RQUOTA_PORT - required only by NFSV3)<br><br>32768 (RQUOTA_PORT - required by both NFSV3 and NFSV4)<br><br>32769 (NLM_PORT - required only by NFSV3)<br><br>**Note:** Make the dynamic ports as static with command **mmnfs configuration change** . | N/A | TCP and UDP | Protocol nodes only |
| Object | swift-proxy-server<br><br>keystone-all<br><br>postgresql-obj | 8080 (proxy server)<br><br>35357 (keystone)<br><br>5000 (keystone public) | 5431 (Object Postgres instance)<br><br>6200-6203 (Object Storage)<br><br>11211 (Memcached) | TCP<br><br>TCP and UDP (for 11211 only) | Protocol nodes only |

## Firewall recommendations for IBM Spectrum Scale GUI

Dedicating certain ports for firewalls helps to secure IBM Spectrum Scale management and install GUIs. Different ports are used for securing installation GUI and management GUI.

The following table lists the ports that need to be used to secure GUI.

*Table 67. Firewall recommendations for GUI*

| Port Number | GUI Type | Protocol |
|---|---|---|
| 9080 | Installation | HTTP |
| 9443 | Installation | HTTPS |
| 80 | Management | HTTP |
| 443 | Management | HTTPS |

The management GUI uses ZIMon to collect performance data. ZIMon collectors are normally deployed with the management GUI and sometimes on other systems in a federated configuration.

Each ZIMon collector uses three ports, which can be configured in ZIMonCollector.cfg. The default ports are 4739, 9085, and 9084.

The port 4444 can be accessible only from the localhost.

## Firewall recommendations for Performance Monitoring tool

The IBM Spectrum Scale system uses the following ports for the Performance Monitoring tool to work.

*Table 68. Recommended port numbers that can be used for Performance Monitoring tool*

| Port Number | Protocol | Service Name | Components that are involved in communication |
|---|---|---|---|
| 4739 | TCP and UDP | Performance Monitoring tool | Intra-cluster |
| 8123 | TCP | Object Metric collection | Intra-cluster |
| 8124 | TCP | Object Metric collection | Intra-cluster |
| 8125 | UDP | Object Metric collection | Intra-cluster |
| 8126 | TCP | Object Metric collection | Intra-cluster |
| 8127 | TCP | Object Metric collection | Intra-cluster |
| 9084 | TCP | Performance Monitoring Tool | Any node that wants to query the database |
| 9085 | TCP | Performance Monitoring Tool | Intra-cluster |

**Important:**

- The 4739 port needs to be open when a collector is installed.
- The 9085 port needs to be open when there are two or more collectors.
- If the 9084 port is closed, accessing the collector to debug or to connect external tools or, even another instance of the GUI, remotely is not possible, except from the node where the GUI and the collector are installed.

## Supported web browser versions and web browser settings for GUI

To access the management GUI, you must ensure that your web browser is supported and has the appropriate settings enabled.

The management GUI supports the following web browsers:

- Mozilla Firefox 41

- Mozilla Firefox Extended Support Release (ESR) 38
- Microsoft Internet Explorer (IE) 10 and 11
- Google Chrome 45

IBM supports higher versions of the browsers if the vendors do not remove or disable function that the product relies upon. For browser levels higher than the versions that are certified with the product, customer support accepts usage-related and defect-related service requests. If the support center cannot re-create the issue, support might request the client to re-create the problem on a certified browser version. Defects are not accepted for cosmetic differences between browsers or browser versions that do not affect the functional behavior of the product. If a problem is identified in the product, defects are accepted. If a problem is identified with the browser, IBM might investigate potential solutions or work-arounds that the client can implement until a permanent solution becomes available.

To configure your web browser, follow these steps:

1. Enable JavaScript for your web browser.

   For Mozilla Firefox, JavaScript is enabled by default and requires no additional configuration.

   For Microsoft Internet Explorer (IE) running on Microsoft Windows 7:

   a. In Internet Explorer, click **Tools** > **Internet Options**.
   b. Click **Security Settings**.
   c. Click **Internet** to choose the Internet zone.
   d. Click **Custom Level**.
   e. Scroll down to the **Scripting** section, and then in **Active Scripting**, click **Enable**.
   f. Click **OK** to close **Security Settings**.
   g. Click **Yes** to confirm the change for the zone.
   h. Click **OK** to close **Internet Options**.
   i. Refresh your browser.

   For Microsoft Internet Explorer (IE) running on Microsoft Windows Server 2008:

   a. In Internet Explorer, click **Tools** > **Internet Options**.
   b. Click **Security**.
   c. Click **Trusted sites**.
   d. On the **Trusted sites** dialog, verify that the web address for the management GUI is correct and click **Add**.
   e. Verify that the correct web address was added to the **Trusted sites** dialog.
   f. Click **Close** on the **Trusted sites** dialog.
   g. Click **OK**.
   h. Refresh your browser.

   For Google Chrome:

   a. On the menu bar in the Google Chrome browser window, click **Settings**.
   b. Click **Show advanced settings**.
   c. In the **Privacy** section, click **Content settings**.
   d. In the **JavaScript** section, select **Allow all sites to run JavaScript**.
   e. Click **OK**.
   f. Refresh your browser.

2. Enable cookies in your web browser.

   For Mozilla Firefox:

   a. On the menu bar in the Firefox browser window, click **Tools** > **Options**.
   b. On the Options window, select **Privacy**.

c. Set "Firefox will" to **Use custom settings for history**.
d. Select **Accept cookies from sites** to enable cookies.
e. Click **OK**.
f. Refresh the browser.

For Microsoft Internet Explorer:

a. In Internet Explorer, click **Tools** > **Internet Options**.
b. Click **Privacy**. Under **Settings**, move the slider to the bottom to allow all cookies.
c. Click **OK**.
d. Refresh your browser.

For Google Chrome:

a. On the menu bar in the Google Chrome browser window, click **Settings**.
b. Click **Show advanced settings**.
c. In the **Privacy** section, click **Content settings**.
d. In the **Cookies** section, select **Allow local data to be set**.
e. Click **OK**.
f. Refresh your browser.

3. Enable file download on IE 10 and 11 running on Windows 2012.
   a. In Internet Explorer, click **Tools** > **Internet Options**.
   b. On the Internet Options window, select the **Security** tab.
   c. On the **Security** tab, click the **Internet zone**.
   d. Click **Custom level** to customize the security level for this zone.
   e. Scroll down to **Downloads** and select **Enable** under File download.
   f. Click **OK**.
   g. Click **Yes** to confirm.
   h. Click **OK** to close the Internet Options.
4. Enable scripts to disable or replace context menus. (Mozilla Firefox only).

   For Mozilla Firefox:

   a. On the menu bar in the Firefox browser window, click **Tools** > **Options**.
   b. On the Options window, select **Content**.
   c. Click **Advanced** by the **Enable JavaScript** setting.
   d. Select **Disable or replace context menus**.
   e. Click **OK** to close the Advanced window.
   f. Click **OK** to close the Options window.
   g. Refresh your browser.

# Chapter 40. GUI limitations

The following are the limitations of the IBM Spectrum Scale GUI:

1. The GUI supports only RHEL7.x or SLES12 as the operating system on Power (Big or Little Endian) or Intel x86. However, other nodes in the IBM Spectrum Scale cluster could be on other platforms and operating systems.
2. Up to 1000 nodes are supported.
3. The GUI supports a subset of the CLI functionality. Additional capabilities will be added in the future releases of the product.
4. The Object management panels do not support configurations with Keystone V2 API, HTTPS communication to Keystone, and AD/LDAP-backed Keystone configurations.
5. One GUI instance supports a single cluster.
6. The GUI does not support file system creation. You need to use the CLI to create a file system.
7. Snapshots are deleted only once in a day to avoid performance issues. Due to this limitation, it is possible that there might be certain discrepancy between the number of snapshots present on the system and the number that should remain according to the snapshot retention rules.
8. In an IBM Spectrum Scale and Elastic Storage Server (ESS) mixed support environment, the ESS GUI must manage the whole cluster to display the ESS-specific pages in the GUI.
9. The GUI does not display performance charts in Internet Explorer. It is recommended to use either Mozilla Firefox or Google Chrome to access GUI in the 4.2.1 release.
10. Users who do not have the *Security Administrator* role cannot modify their own password. If such users want to change their password, they can request the user with *Security Administrator* role to expire their password by selecting the **Expire Password** option from the GUI. This causes the system to prompt the user to change the password at the next login.

For limitations of the installation GUI, see *Installing IBM Spectrum Scale by using the graphical user interface (GUI)* in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

# Accessibility features for IBM Spectrum Scale

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

## Accessibility features

The following list includes the major accessibility features in IBM Spectrum Scale:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are discernible by touch but do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

IBM Knowledge Center, and its related publications, are accessibility-enabled. The accessibility features are described in IBM Knowledge Center (www.ibm.com/support/knowledgecenter).

## Keyboard navigation

This product uses standard Microsoft Windows navigation keys.

## IBM and accessibility

See the IBM Human Ability and Accessibility Center (www.ibm.com/able) for more information about the commitment that IBM has to accessibility.

# Notices

This information was developed for products and services 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.

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,

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



## A

# B



# C

# D



# E

# F

# G



# H

# I

# J



# K



# L



# M

# N

# O

# P

# Q



# R

# S

# T



# U

# V



# W



# Y

Product Number: 5725-Q01
5641-GPF
5725-S28

Printed in USA

SA23-1455-01